



Steve Mann

ייעוץ מקצועי:
שרון טל

<?xml version="1.0"?>

<do type="accept">

<wml>

WAP

למפתחי אתרים באינטרנט



WAP
למפתחי אתרים באינטרנט

הוראות התקנה והפעלה של התקליטור
בנספח ה' ובקובץ ONCD בתקליטור

עורך ראשי: **זהר עמיהוד**

ייעוץ מקצועי וכתיבת פרקים 2, 3 ו- 7: **שרון טל**

תרגום: **אבי מילשטיין**

עריכה לשונית ועיצוב: **ענבל אילני, שרה עמיהוד**

עיצוב עטיפה: **שרון רז**

שמות מסחריים

שמות המוצרים והשירותים המוזכרים בספר הינם שמות מסחריים רשומים של החברות שלהם. הוצאת הוד-עמי והוצאת Wiley עשו כמיטב יכולתן למסור מידע אודות השמות המסחריים המוזכרים בספר זה ולציין את שמות החברות, המוצרים והשירותים. שמות מסחריים רשומים (registered trademarks) המוזכרים בספר צוינו בהתאמה.

הודעה

ספר זה מיועד לתת מידע אודות מוצרים שונים. נעשו מאמצים רבים לגרום לכך שהספר יהיה שלם ואמין ככל שניתן, אך אין משתמעת מכך כל אחריות שהיא.

המידע ניתן "כמות שהוא" ("as is"). הוצאת הוד-עמי והוצאת Wiley אינן אחראיות כלפי יחיד או ארגון עבור כל אובדן או נזק אשר ייגרם, אם ייגרם, מהמידע שבספר זה, או מהתקליטור המצורף.

לשם שטף הקריאה כתוב ספר זה בלשון זכר בלבד. ספר זה מיועד לגברים ונשים כאחד ואין בכוונתנו להפלות או לפגוע בציבור המשתמשים/ות.

- **טלפון: 09-9564716**
- **פקס: 09-9571582**
- **דואר אלקטרוני: info@hod-ami.co.il**
- **אתר באינטרנט: www.hod-ami.co.il**

WAP
למפתחי אתרים באינטרנט

Steve Mann





Programming Applications with the Wireless Application Protocol (WAP)

By Steve Mann

Editor: **Z. Amihud**

Authorized translation from the English language edition published by John Wiley & Sons, Inc.

הוצאת הוד-עמי

לספרי מחשבים בע"מ

ת.ד. 6108 הרצליה 46160

טלפון: 09-9564716 פקס: 09-9571582

info@hod-ami.co.il

הודפס בישראל 2001

HOD-AMI Ltd.
P.O.B. 6108, Herzliya
ISRAEL, 2001

ISBN 965-361-282-4 מסת"ב

תוכן עניינים מקוצר

תוכן העניינים

פרק 4: WML 47

הקדמה

"על שלושה דברים איש ההי-טק עומד: Startup, Internet ו-WAP". הוצאת הוד-עמי שמחה לתמוך בך, מפתח התוכן מבוסס ה-Web, במאמציך להביא את המידע והיישומים שלך אל מסד הובלת המידע החדש והיותר מלהיב מאז ראשית ה-Web. קוד המקור שבתקליטור המצורף לספר זה, יאפשר לך לתרגל מיידית את מה שלמדת בספר, ולהביא תוכן משלך למסכי הטלפונים הסלולרים ברחבי העולם.

הטלפונים הסלולרים משנים במהירות את צורת התקשורת של בני האדם בישראל וברחבי העולם.

אנליסטים של התעשייה צופים שיותר ממיליארד טלפונים סלולרים יפעלו בשנת 2003. WAP Forum נוסד בשנת 1997 על ידי Phone.com, Ericsson, Motorola ו-Nokia. הוא קבוצה תעשייתית ששמה לה למטרה לאפשר טלפוניה ושירותי מידע מתוחכמים באמצעות מכשירים סלולרים ידניים. WAP (**W**ireless **A**pplication **P**rotocol) הוא למעשה הסטנדרט העולמי למידע סלולרי ושירותי טלפוניה עבור טלפונים דיגיטלים ניידים ומסופים סלולרים אחרים. יצרני הטלפונים הסלולרים המייצגים יותר מ-95% מהשוק העולמי בכל הסטנדרטים הדיגיטליים התחייבו לשווק את מכשיריהם עם תמיכה ב-WAP. ספקי שירות המייצגים מאות מיליונים של מנויים ברחבי העולם הצטרפו ל-WAP.

היכרות

ברחבי העולם קיימים יותר מ- 300 מיליון משתמשים בטלפונים סלולרים. אליהם מצטרפים מיליוני המשתמשים באינטרנט בעולם. שני שווקים אלה מתכנסים יחד אל אותה נקודה: מכשירים נישאים ממוחשבים קטנים. מכשירים אלה צריכים להתאים לתקשורת קולית באיכות גבוהה, לתקשורת נתונים ברוחב פס מתאים (5 עד 10Kbps), קישוריות אינטרנט חלקה ולגישה לשירותי אינטרנט כגון e-mail ותוכן. בנוסף לכך הם צריכים להיות בעלי יכולת תכנות ולהיות מסוגלים להפעיל יישומים אישיים. כדי לפשט את התיאור, נקרא למכשירים אלה - טלפונים חכמים.

WAP Forum הוקם על ידי Ericcson, Motorola, Nokia ו-Phone.com (לפנים Unwired Planet). הן החלו לשתף פעולה בפיתוח ארכיטקטורה מתקדמת לשירותי העברת נתונים לטלפונים חכמים. ארכיטקטורה זו, השואבת רבות מטכנולוגיות אינטרנט קיימות, מבוססת על פרוטוקול הנקרא Wireless Application Protocol (WAP).

WAP נמצא בשימושם של ספקי שירותים סלולרים רבים. הוא מספק שירותים כגון דחיסה, הצפנה, שילוב של שירותי נתונים וטלפוניה וחשוב אף יותר – WAP משמש לבניית יישומים הפועלים על טלפונים חכמים – יישומי WAP.

יישומי WAP הם המפתח להפיכת טלפונים חכמים לחכמים באמת. חומרה איננה פותרת באמת בעיות משתמש, או מספקת למשתמשים יכולות אינטליגנטיות מתקדמות – התוכנה היא שעושה זאת. לסיכום, אנו שמחים להציג לפניך את הספר **WAP למפתחי אתרים באינטרנט**.

מפת דרכים

בספר זה שמונה פרקים:

פרק 1 "היכרות עם WAP", מתווה את הכיוון של הספר. הפרק מתאר את השווקים והטכנולוגיות שאליהם מכוון WAP וכולל סקירה על רוב רכיבי WAE (**W**AP **A**pplication **E**nvironment). הפרק מספק תשובה לשאלה החשובה "מדוע WAP?" ובנוסף מתאר את הדמיון והשוני בין פיתוח יישומים ל-Web ול- WAP.

פרק 2 "כלים והגדרה", עוסק בכלי הפיתוח השונים העומדים לרשות המפתחים. הפרק מציג את כלי הפיתוח של Nokia, עימו ניתן לעבוד במחשב שלך במצב Off line, ואת אתר ccwap, עימו ניתן להריץ את קבצי WML במצב On line. כמו כן, הפרק מנחה אותך בתהליך המרת תמונות לפורמט wbmp, המוכר כתקן לפיתוח יישומי WAP.

פרק 3 "הגדרת PWS כשרת WAP", מתחיל משלב התקנת PWS, דרך הגדרת סוגי הקבצים הנדרשים לעבודה עם WAP (MIME type) ועד לבדיקה שההתקנה בוצעה כראוי.

פרק 4 "WML", מתאר את ה- Wireless Markup Language, כלי התכנות העיקרי ליצירת יישומי WAP. WML היא שפה תואמת XML בעלת תחביר מתויג בדומה ל-HTML. אתה יוצר יישומי WAP על ידי יצירת מסמכי WML, סטטיים או דינמיים, ומסירתם להתקני WAP. בפרק נכללים קטעי קוד קצרים כדוגמאות.

פרק 5 "WMLScript", עוסק בשפת התסריט המהווה חלק של WAE. WMLScript מוסיפה יכולות שגרה קלות משקל וספריות פונקציה ל-WML. יחד הן מספקות לך סביבת פיתוח יישום עשירה וחזקה.

פרק 6 "i18N (בינלאומיות ועברית)", מתאר נושאים שיש לשים לב אליהם בעת בניית יישומי WAP, האמורים לפעול בכל רחבי העולם ובעברית בפרט. כסטנדרט גלובלי שפותח על ידי קבוצה בינלאומית של חברות, עבודתו של WAP forum מתמקדת בפתרונות בינלאומיים יעילים. פרק זה מציג לפניך קבוצות תווים, קידוד תו ושפות ואת השפעתם על תפקוד WAP.

פרק 7 ״יצירת יישומי WAP ו-ASP״, עוסק ביצירת דפי WML בצורה דינמית באמצעות ASP. הפרק מציג דוגמה לבניית קטלוג המקושר למסד נתונים של Access (קובץ mdb) הנמצא בשרת.

פרק 8 ״Caching (מיטמון)״ מתאר טכניקות מיטוב של יישומי WAP, על ידי ניהול המטמון שבהתקן WAP. יישומים המשתמשים ברשתות סלולריות חייבים להיות קטנים ומהירים. הדרך הטובה להשגת מטרות אלו היא שימוש מועט ככל הניתן ברשתות. מיטמון הוא המפתח.

הוספנו מספר נספחים יעילים המהווים מדריך מהיר ל- WML ולאפיוני WAP אחרים. בקריאה ראשונה ייתכן ש- WML תהיה מעט מבלבלת, נספחים א׳ ("WML Elements"), ב׳ ("WML Elements Cross Reference") ו- ג׳ ("WML Attributes Cross Reference") מהווים כלים תמציתיים להבנת מהותה של WML. נספח ד׳ ״WMLScript Library Function״, מהווה גם הוא סיכום תמציתי של כל ספריות WMLScript. ארבעת נספחים אלה הם כלי התחלה מהירה ליצירת יישומי WAP.

למי מיועד ספר זה?

ספר זה מיועד בראש ובראשונה למפתחי יישומים. פיתוח יישומים סלולרים משלב הבנת תכנות מסורתי בשפות הדור השלישי (Visual Basic, C++), פיתוח תוכן Web תוך שימוש בשפת HTML ובשפות תסריט (VBScript, JavaScript) והבנת דרך פעולת HTTP. אנו מניחים שאתה יודע לתכנת. אם אתה שולט ב- C, C++, Basic, Java או במספר שפות תכנות מודרניות אחרות, יודע HTML ולפחות שפת תסריט אחת (JavaScript או VBScript) תוכל לקרוא ספר זה ולהבינו ללא קשיים מרובים.

לעומת זאת, לא הנחנו שפיתוח יישומי Web בסביבת WAP מוכר לך. היכן שנדרש, אנו מסבירים את הדרוש הסבר: WAP, HTTP, MIME ועוד. ספר זה אינו הסבר מפורט על כל האינטרנט – אלא רק הדברים הדרושים לך כדי לבנות יישומי WAP.

למרות התמקדותנו ביישומי WAP, ספר זה מתאים גם למנהלים טכניים שדרושה להם סקירה על האפשרויות הממשיות של WAP. אנו דנים ברמתה הגבוהה של הארכיטקטורה, יותר מאשר בפריטים שרמתם נמוכה יותר. ספר זה יהיה לתועלת גם למפתחי תוכן Web, אפילו אם אינם יודעים רבות אודות תכנות. במובן מסוים, WAP הוא רק כלי הובלה של מסמכים לדפדפן בעזרת WML ו-WMLScript במקום בעזרת HTML ו-JavaScript.

בספר זה ניסינו להתמקד בפרטים המעשיים והמועילים למפתחי יישומים, ולכן כללנו בו קטעי קוד מועילים רבים.

התקליטור המצורף

כל קטעי הקוד המופיעים בספר זה, נמצאים בתקליטור המצורף. תוכל להשתמש בקטעים אלה בשלמותם כדי לחסוך זמן, ולמנוע טעויות הקלדה במהלך העבודה עם הדוגמאות שבספר.

התיקיה הרלוונטית לספר זה היא **X:\Books\59313**, בה נמצאות תיקיות משנה לפי פרקי הספר. תוכל להעתיק את התיקיה לדיסק הקשיח שלך **ולא** לשכוח לבטל את הסימון **קריאה בלבד** ממאפייני התיקיה ו/או הקובץ.

להסבר נוסף פנה לנספח ה' ולקובץ ONCD שבתקליטור.

איך מריצים את הקוד?

כדי להריץ את קוד המקור הנלמד בספר שנמצא בתקליטור, יהיה עליך להוריד מהאינטרנט את הערכה של Nokia ו/או של חברה אחרת, לפי ההנחיות שבפרק 2.

על היועץ המקצועי

שרון טל, בעל תואר BSc. במדעי המחשב. מדריך בקורסי Web Master, Web Programming, Visual C++ ו-Java במרכזי הדרכה מובילים בישראל. משמש כמנהל פיתוח בחברת d-Say Interactive. יועץ בתחום הסלולר לחברות הי-טק בארץ ובחו"ל.

לשאלות מקצועיות על הספר תוכלו לפנות: sharon@d-say.com

הערות, שאלות, רעיונות

לא נחסך כל מאמץ להבטחת הדיוק של ספר זה והתקליטור המצורף אליו. אם יש לך הערות, שאלות או רעיונות הנוגעים לספר זה של ההוצאה ולתקליטור המצורף, אנא שלח אותם להוצאת הוד-עמי באחת השיטות הבאות:

דואר אלקטרוני: info@hod-ami.co.il, בשורת Subject ציין את המספר **59313**

דואר רגיל:

הוצאת הוד-עמי לספרי מחשבים

ת.ד. 6108

הרצליה 46160

פרק 1

היכרות עם WAP

ברחבי העולם קיימים כיום יותר מ-300 מיליון משתמשים בטלפון סלולרי ומיליוני משתמשים באינטרנט. לדעת כל המומחים, מספר המשתמשים בשתי טכנולוגיות אלו ימשיך ויגדל במהירות במשך מספר שנים.

טלפונים סלולרים מתאימים ביותר לקיום קשר עם שאר העולם (בהנחה שאתה בטווח הכיסוי, אך זהו כבר סיפור אחר). הם קטנים, קלי משקל ואינם יקרים. ברוב הארצות קיימת רשת תקשורת סלולרית אחת או יותר. ארצות מפותחות פחות, גם הן בונות תשתיות לטלפון סלולרי, מכיון שקל ומהיר יותר להניח תשתית של רשת סלולרית מאשר תשתית של רשת קווית רגילה. הכיוון הנוכחי מורה על כך שהטלפוניה הסלולרית תהפוך לצורה העיקרית של תקשורת קולית בעולם.

הגישה לאינטרנט משנה דרכי פעולה והתנהגות, כאשר e-mail ו-Web הופכים להיות השפה המוכרת של החיים המודרניים: בני משפחה הנמצאים בריחוק פיסי משתמשים ב- e-mail כדי לשמור על קשר, עסקים משתמשים ב-Web כדי לשווק ולמכור מוצרים, באפשרותך לקבל חדשות ממוקדות ישירות למחשב שלך, ניתן למכור חפצים אישיים באחד מאתרי המכירות הפומביות הרבים שברשת, ואפילו לנהל מכירת מטלטלים וחפצים אישיים באופן וירטואלי. ביכולתך לקנות ולמכור סחורה, לבדוק לוחות זמנים של טיסות, לקבל מידע אודות מזג האוויר ברחבי העולם, וגם לקבל את רשימת הסרטים בבית הקולנוע השכונתי, להזמין פיצה, לשלוח פרחים... ומה לא? למעשה, יש עשרות אלפי שימושים באינטרנט.

טלפוניה סלולרית היא טכנולוגיה ניידת. תוכל לשאת את הטלפון אתך, לשדר ולקבל קריאות מכל רחבי העולם. נכון להיום, הגישה לאינטרנט היא בעיקרה פעולת מחשב המתבצעת באמצעות מודם וקו טלפון ממשי. הגישה לאינטרנט הנייד היא חלום שטרם מומש עדיין. למרות שניתן להשיג גישה לאינטרנט אלחוטי, באמצעות שימוש במודם אלחוטי ומחשב נישא, או באמצעות מחשב כף-יד ואיתורית דו-כיוונית, פתרונות אלה אינם אידיאלים מסיבות רבות, כמו גודל ההתקן, משקל, חיי הסוללה ועלות.

מספר מגמות מביאות להתכנסות של הטכנולוגיות הסלולריות עם טכנולוגיות האינטרנט:

- יורדת עלות הקנייה והשימוש בטלפונים סלולרים. באזורים רבים בעולם תקשורת סלולרית זולה יותר מאשר תקשורת קווית רגילה. כתוצאה מכך, השימוש בטלפון הסלולרי הופך לנפוץ יותר.

- טכנולוגיית המחשב והטכנולוגיה הסלולרית גם יחד הן בעלות עוצמה הולכת וגדלה. אנו עדים לשיפורים עצומים בזיכרון, בסוללה וביחידות המעבד, המאפשרים הן ליצרני הטלפונים והן ליצרני המחשבים לספק עוצמה משמעותית נוספת בעלות נמוכה יותר.
- השימוש באינטרנט גדל. אנשים רבים יותר מכירים בערך העצום של השימוש באינטרנט, הן לגבי תקשורת והן לגבי תוכן. לכן, טבעי לרצות בגישה מתמדת לאמצעי זה ללא צורך בנשיאת מחשב ומבלי להתעמק בבעיות המסובכות של הוספת יכולות אלחוטיות למחשב זה.
- אנשים רוצים מידע כאשר הם זקוקים לו, ולא רק כאשר הם נמצאים ליד המחשב השולחני. הטלפון הסלולרי הוא המכשיר היותר מתאים לקבלת מידע מיידי בכל עת. כמו כן, יותר מ-300 מיליון בני-אדם כבר נושאים טלפון כזה, מיליוני משתמשים חדשים נוספים בכל שנה וכיום ניתן להתקשר כמעט לכל אחד.
- הטלפונים הסלולרים "חכמים" יותר מבעבר. הם אינם מכשיר להולכת תקשורת קולית בלבד. יצרנים החלו להוסיף לטלפון זיכרון נוסף ומעבדים חזקים יותר, וכך ניתן לכלול בו יישומים מגוונים, כגון תוכנות ניהול תוכן, רשימת משימות לביצוע ותוכנות אישיות מועילות נוספות.
- יכולתן של הרשתות הסלולריות הולכת וגדלה. כבר מספר שנים ניתן להשתמש ברשתות סלולריות אנלוגיות לתקשורת נתונים (גם אם בקשיים מסוימים). הרשתות הסלולריות הדיגיטליות החדשות מתאימות הרבה יותר הן לתעבורת קול והן לתעבורת נתונים.
- השימוש באינטרנט ובטלפונים סלולרים מתכנס לקראת מחיר יציב וקבוע. שני השווקים נהגו לחייב לפי מספר דקות השימוש (טלפונים סלולרים) או לפי שעות (אינטרנט). מחיר קבוע ותמחור אינטרנט לא מוגבל הוא הנורמה כעת. מחיר קבוע לתעבורה סלולרית קולית ולתעבורת נתונים מקובל כעת בארה"ב; מזה מספר שנים נוהג זה מקובל גם במקומות אחרים בעולם עבור שירותים מסוימים. עלות קבועה מקלה על הצרכן הפוטנציאלי של נתונים סלולרים להעריך מראש את ההוצאות.

כתוצאה ממגמות אלו, יצרני טלפונים סלולרים מתחילים לייצר טלפונים "חכמים", שיש להם יכולת תכנות מלאה המוטבעת בחומרה שלהם. יצרני התקני מחשב מתחילים לבנות מחשבי כף-יד (Visor, Palm Pilot, PocketPC וכדומה) בעלי יכולות תקשורת סלולרית. שני השווקים מתכנסים לאותה נקודה: התקנים ממוחשבים קטנים, קלי משקל, המתאימים באותה מידה לתקשורת קולית עם איכות גבוהה, תקשורת נתונים בעלת רוחב פס בינוני (5-10Kbps), קישוריות נוחה ופשוטה לאינטרנט, גישה לשירותי אינטרנט כגון e-mail ותוכן ולשימוש כללי. בעתיד נראה יותר התקנים ממוחשבים הניתנים לתכנות, שיכולים להריץ יישומים אישיים.

WAP Forum

בראשית שנות ה-90, התברר ליצרני הטלפונים הסלולרים הגדולים שקול, נתונים סלולרים ואינטרנט יכולים להתכנס. הם החלו לפתח טכנולוגיות, כדי להאיץ את מגמה הזו. בה בעת, חברת Start-up מעמק בסיליקון, שנקראה לפנים Unwired Planet וכיום Phone.com, החלה גם היא לפתח טכנולוגיות שתאפשרנה התכנסות זו.

שפת HDML (**H**andheld **D**evice **M**arkup **L**anguage) של Phone.com שימשה כבסיס ל**שפת הסימון הסלולרית** – **WML** (**W**ireless **M**arkup **L**anguage). החברה, שמרכזה בעמק הסיליקון ויש לה פעילויות ביפן ובאנגליה, היא כעת ספק מוכר של מיקרו-דפדפנים תואמי WAP ו-WAP Gateway Technologies. מוצרי Phone.com נמצאים בשימוש יצרני המכשירים הניידים המובילים בעולם, בכללם AT&T Wireless, Bell Atlantic-Mobile, GTE Wireless, Mitsubishi, Panasonic, Qualcomm, Samsung, Siemens ואחרים.

ביוני 1997 הכריזו שלושת היצרניות הגדולות של טלפונים סלולרים – Ericsson, Motorola ו- Nokia – יחד עם Unwried Planet (היום Phone.com) על **WAP Forum** (**W**ireless **A**pplication **P**rotocol). זהו ארגון ללא כוונת רווח שמטרתו הגדרת תקנים לאספקת גישה לאינטרנט עבור התקנים סלולרים לשימוש אישי. הפורום פתוח לכל מי שמעוניין, ובמיוחד למפתחי תוכן, יצרני התקנים, ספקי שירות וספקי תשתית. מטרותיו של פורום WAP :

- טיפוח אספקת שירותי תוכן אינטרנט ונתונים מתקדמים לטלפונים סלולרים והתקנים סלולרים ניידים אחרים. שירותים אלה יכולים להיות שירותים בסיסיים פשוטים, כגון e-mail סלולרי וגישה ל- Web, ובנוסף לכך כוללים פתרונות ייחודיים עבור שווקים וירטואליים, כגון שידור ושירותי שדה.
- יצירת הגדרות פרוטוקול גלובלי הפועל על הרשתות הסלולריות בכל רחבי העולם. תקן WAP מכיל הגדרות של מחסנית פרוטוקול, הניתנות להתאמה קלה ל**שירות הודעות קצרות** – **SMS** (**S**hort **M**assage **S**ervice), ל-**USSD** (GSM **U**nstructured **S**upplementary **S**ervice **D**ata), ל-**CDPD** (**C**ellular **D**igital **P**acket **D**ata) ולרשתות אחרות.
- אפשור יצירת תוכן ויישומים שיפעלו בטווח רחב של התקנים ורשתות. **WAE** (**W**AP **A**pplication **E**nvironment) מכילה הגדרות ליישומי ממשק משתמש (UI) עצמיים, היכולים לפעול ללא שינוי על מיגוון התקנים בעלי מאפיינים שונים.
- אימוץ והרחבה של תקנים וטכנולוגיות קיימים היכן שניתן ומתאים. אם לדייק, פורום WAP אימץ רבים מתקני האינטרנט הקיימים, כדי לתת פתרון לבעיות הייחודיות של רשתות סלולריות, הכולל התמקדות מיוחדת בהתאמה של טכנולוגיות ויישומי WAP להפעלה בהיקף בינלאומי.

במטרה להשיג מטרות אלו, פורום WAP צריך לשקול בזהירות את המאפיינים של התקני המטרה וסביבת הסלולר שהם פועלים בה. למרות הגידול ביכולות של הטלפונים הסלולרים, הם עדיין נמכרים בשוק קונים שיש בו רגישות-מחיר גבוהה. כל דולר של עלות ייצור הניתן להיחסך הוא קריטי.

מאפיינים של התקני WAP

פורום WAP הניח הנחות מסוימות אודות התקן WAP טיפוסי בשעה שפעל להגדרת מאפייני היסוד. הפורום לא חיבר הוראות עיצוב כלשהן, כפי ש-Microsoft עשתה עבור מכשירים לשימוש אישי שבהם מופעלת מערכת Windows CE (כמו לדוגמה, הכתבת מינימום RAM ו-ROM, מספר יציאות התקשורת ומנגנון כניסת הנתונים). במקום זאת, בהתבסס על מיגוון הטלפונים הסלולרים הנוכחי, איתוריות דו-כיווניות ומחשבי כף-יד, הפורום הגדיר קבוצת מאפיינים להתקנים דומים.

- להתקן WAP יש רכיבים כמו למחשב אישי: יע״מ (CPU – יחידת עיבוד מרכזית), זיכרון לגישה אקראית (RAM) וזיכרון לקריאה בלבד (ROM). עם זאת, הן הרכיבים והן מהירות עיבוד מוגבלים ביכולותיהם. אין כללים נוקשים נוספים ביחס למפרטי החומרה. האמצעים הנדרשים **מוגבלים** ומספיקים לביצוע העבודה, אך לא יותר.
- מכיון שהתקני WAP הם התקנים סלולרים ניידים, אורך חיי הסוללה שלהם סופי ומוגבל. גם כאן אין כללים נוקשים: אם תבחן את הטלפונים הסלולרים הנוכחיים, תמצא שזמן הדיבור נע בין שעתיים ל- 10 שעות. למרות שטכנולוגיית הסוללות משתפרת, אין לצפות להכפלת היכולת בכל 18 חודש, כפי שצופה חוק Moore ביחס ליחידות עיבוד מרכזיות של מחשבים. הנקודה החשובה היא שיצרנים ומפתחי יישומים יעשו כל שביכולתם כדי לשמר את חיי הסוללה על ידי כתיבת תוכניות קצרות, מהירות ויעילות, ועל ידי שימוש בטכניקות שימור מתח יעילות אחרות. הכוונה היא שייעשה שימוש מצומצם ככל האפשר ברשת. אמצעים סלולרים להעברת הודעות צורכים כוח סוללה רב, הרבה יותר מאשר מחשבים ישנים רגילים.
- רשתות WAP פועלות במתח-נמוך ובעלות רוחב פס בינוני של פחות מ- 10Kbps (נכון להיום). הדבר קשור לבעיות חיי סוללה – ככל שרוחב הפס גדל, גדלה צריכת המתח. ככל שגדלה צריכת המתח מתקצרים חיי הסוללה, כפי שמראה 3COM עם מחשבי כף-היד הפופולריים שלה. משתמשים מעדיפים כמובן להחליף את הסוללות לעיתים רחוקות ככל הניתן.
- רשתות סלולריות לקול ונתונים הן ביסודן לא אמינות, לא יציבות ובלתי צפויות. פרוטוקול WAP והיישומים הפועלים על התקני WAP חייבים להיות יציבים ואמינים. עליהם להיות מסוגלים להתמודד עם הפרעות בשידור, שבירת קשר ואבדן שירות. למרבה המזל, מעצבי פרוטוקול WAP טיפלו בבעיות אלו, כך שמעצבי יישומים אינם צריכים לעשות זאת.

- התקני WAP יהיו מטבעם בעלי מסכים קטנים ויכולת הוספת נתונים מוגבלת. זהו תוצר לוואי טבעי להקטנת עלויות הייצור ושמירת גודל ההתקן הסלולרי. ככל שהמסך קטן יותר, כן קטנה העלות. אם מוותרים על לוח מקשים מלא ובמקום זאת משתמשים במקלדת המקובלת היום בטלפונים הסלולרים, שבה כל מקש מייצג אותיות אחדות, מוזילים את העלות במידה נוספת. עד כה, כל התקני WAP שהוכרזו או שווקו היו למעשה טלפונים סלולרים עם מספר מקשים נוספים ומספר תכונות מבוססות-תוכנה נוספות. היום מקובלים מסכים קטנים בגודל של ארבע שורות בנות 12 תווים וקלט באמצעות מקלדת טלפון רגילה, הכוללת ביסודה את מערכת הספרות. יש לצפות בדגמים העתידיים לתצוגות גדולות יותר, לוח מקשים המבוססים על מסך-מגע, זיהוי קול ושיפורים נוספים. בה בעת, יש לצפות שהדברים יישארו קטנים, פשוטים וזעירים. ראוי לציין כאן את ה-Palm שהזנת הקלט שלו נעשית באופן שונה, ואת השילוב שלו במכשיר pdQ של Qualcomm.

בנוסף לשיקולים של הגדרת מאפייני ההתקן, פורום WAP מתמקד בכישורי המשתמש. השיקול העיקרי הוא טבען של הרשתות הסלולריות. ברשתות סלולריות יש זמן השהיה (latency) ארוך. המשמעות היא שיש הפסקה משמעותית בין זמן הצגת הדרישה לפעולה על ידי המשתמש והשידור שלה על ידי ההתקן ברשת הסלולרית, ועד התגובה המגיעה להתקן לצפיית המשתמש. בעת שימוש במכשיר הסלולרי מציע WAP למשתמש ליצור דרישות ממוקדות וקטנות של גושי נתונים קטנים (בדרך כלל, עד 1000 בתים) – כמו מספרי טלפון, כתובות, רשימת סרטים, רשימת מסעדות, זמני המראה ונחיתה של מטוסים וכדומה – ולא דפדוף מקרי באתרי Web כפי שנוהגים במחשב שולחני. ככל שהדרישה והתגובה יהיו תמציתיות יותר, כן תקטן ההשהיה.

עיצובים של טלפונים סלולרים מסוימים מסובכים מדי וכוללים תכונות מיותרות, שלא משתמשים בהן. בנוסף לכך, משתמש טיפוסי בטלפון סלולרי נע ממקום למקום ועוסק במספר דברים באופן סימולטני (כגון: נהיגה במכונית, דיבור בטלפון נייד המחובר לדיבורית והאזנה לרדיו). אין סיבה לחשוב שמשתמש בהתקן WAP יהיה שונה. מכיון שלהתקן WAP יש יכולות רבות יותר מאשר לטלפון סלולרי, חשוב שההתקן יהיה קל לשימוש ככל האפשר.

מפרטי WAP

להלן המפרטים של רכיבי המפתח של WAP:

- **מודל תכנות המבוסס בעיקרו על מודל תכנות Web הקיים**. בדומה לדפדפן Web, התקן WAP מעורב בסדרת העברות נפרדות של דרישה/תגובה עם שרתי תוכן, המספקים תוכן סטטי ותוכן דינמי גם יחד. הדבר מאפשר שימוש בידע הנוכחי על דרך פעולת ה- Web וכלי ה- Web הקיימים ליצירת תוכנת WAP.
- **שפת סימון תואמת XML ומעוצבת ליצירת יישומי WAP להתקנים עצמיים.** שפת הסימון לסלולר (WML) מכילה מספר מוגבל של רכיבי XML (שנקראים תגיות ב-HTML). היא מניחה הנחות מעטות בלבד אודות ההתקן שיפעיל את התוכנה.

היישומים מחולקים ליחידות פעולה קטנות שניתן להעבירן דרך הרשת הסלולרית. כמו HTML, גם WML מתמקדת בהצגת תוכן תבניתי ובעלת אפשרויות הפעלה מוגבלות מצד המשתמש.

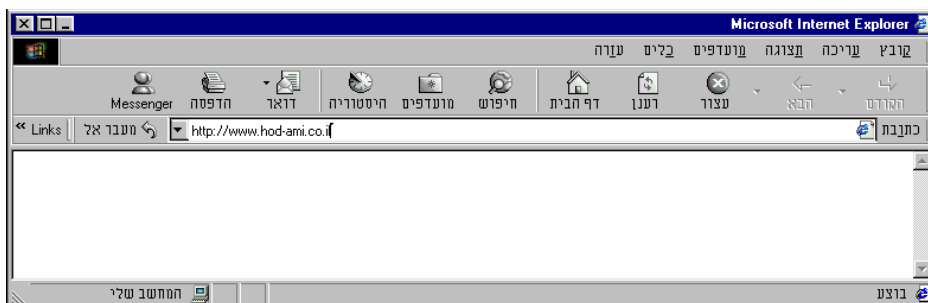
- **שפת התסריט WMLScript מבוססת על ECMAScript של ECMA (אגודת יצרני המחשב האירופאית) ומיועדת להרחבת יכולותיה של WML.** שלא כ-WML המתמקדת במה שרואה המשתמש, ל- WMLScript יש מעט מאוד יכולות ממשק. במקום זאת, היא מעוצבת באופן שמאפשר שיפור יכולות המחשוב של WML.
- **מפרט מיקרו-דפדפן (microbrowser) המגדיר כיצד WML ו-WMLScript אמורות להיתרגם בהתקן הידני ולהיות מוצגות למשתמש.** מפרט זה מציג את האלוגריתמים העיקריים שסוכן-משתמש (User Agent שהוא WAP Device, Client) חייב ליישם. לעומת זאת, הוא אינו קובע את צורת הצגת התוכן. עניין זה נתון לשיקול דעתו של כל יצרן התקנים בנפרד. הוא מגדיר את התפקוד המינימלי ואת ממשק המשתמש הבסיסי שהתקן תואם WAP חייב לכלול. חברי פורום WAP חופשיים להוסיף יכולות נוספות, כדי למלא צרכים מיוחדים של הצרכנים שלהם.
- **מסגרת ליישומי טלפוניה סלולרית (WTA), המספקת פונקציות לשימוש ספקי השירות, כדי לשלב פונקציות טלפון ומיקרו-דפדפנים בהתקן WAP.** לדוגמה, ספקי שירות יכולים לכתוב תוכנה ב-WML כדי לטפל בשיחות נכנסות, אחזור דואר קולי, הפניית הודעות, או שינוי רשימת כתובות של הטלפון. שתי פונקציות WTA בלבד זמינות לכל מפתחי היישומים: ניתן לחייג ולשלוח צלילי DTMF (**D**ual **T**one **M**ulti-**F**requency) במהלך שיחת טלפון.
- **מחסנית פרוטוקול פשוטה המעוצבת להקטנת דרישות רוחב-פס, לעבודה עם אמצעי העברה סלולרים שונים ולאספקת תקשורת בטוחה.** היא כוללת דחיסה אוטומטית של התנועות והתוכן, ובכלל זה תמיכה בהידודיות (אינטראקטיביות) עם פרוטוקולי אינטרנט נפוצים ומיפוי שלהם.

כמפתחי יישומים אינכם צריכים לדעת הרבה אודות מסגרת WTA או אודות מחסנית הפרוטוקול כדי לכתוב תוכנית WAP טובה. עם זאת, עליכם להבין היטב את יסודות מודל היישום ואת מפרטי המיקרו-דפדפן, וגם להכיר את WML ואת WMLScript.

מודל התכנות Web

מודל התכנות WAP מבוסס על מודל התכנות World Wide Web, ובקצרה – Web. במודל Web המשתמש מפעיל דפדפן Web ופונה אל URL (**U**niform **R**esource **L**ocator). תרשים 1.1 מציג כיצד המשתמש עשוי לפנות אל אתר www.hod-ami.co.il תוך שימוש בשדה URL באמצעות הדפדפן Internet Explorer.

דפדפן Web מנתח את הכתובת שקיבל (URL), ושולח לשרת Web דרישת HTTP או HTTPS (**H**yper **T**ext **T**ransfer **P**rotocol **S**ecure).



תרשים 1.1 דרישת URL

הפקודה GET דורשת את מסמך השורש מהשרת: GET/HTTP/1.1

השרת מבקש את מסמך השורש מכיון שהמשתמש אינו מפרט מסמך מסוים בכתובת ששלח. השורש הוא מסמך ברירת המחדל ב-Web.



הערה!

ייתכן שהדפדפן ישלח כמות גדולה של נתונים בנוסף לדרישת GET בתבנית של כותרות HTTP. כותרות אלו יכולות לציין את מספר גרסת הדפדפן, את תבניות התוכן שביכולתו לקבל, את תאריך וזמן הדרישה. כדי לפשט, איננו מראים כאן את הכותרות. ראה בפרק 8 הסברים מפורטים על כותרות HTTP.

שרת Web מקבל ומנתח את הדרישה. אם הדרישה לגיטימית, הוא לוקח את תוכן הקובץ הסטטי או את הפלט של תוכנית CGI (**C**ommon **G**ateway **I**nterface), ומחזיר אותם כחלק מתגובת HTTP. חלק התוכן של התגובה הוא ב-HTML – שפת הסימון והתצוגה המובנת לדפדפני Web. במקרה זה, השרת מאחזר את תוכן הקובץ index.html, ומצרף אותו לתגובת HTTP טיפוסית.



הערה!

שרתי Web מוגדרים להחזרת קובץ ברירת מחדל, אם לא צוין קובץ מיוחד כלשהו. קבצי ברירת המחדל היותר נפוצים הם index.htm ו-index.html, למרות שהקבצים יכולים להיות אחרים.

התגובה מצביעה על סטטוס הדרישה (במקרה זה, OK).

```
HTTP/1.1 200 OK
<html>
  <body>
    <p>WorldFAQ is a web site that provides information about places
       around the world. It is currently under construction.
    </p>
  </body>
</html>
```

הדפדפן מקבל את ההודעה, מנתח את גוף ההודעה ומציג את תוכן הקובץ index.html על מסך הדפדפן בתבנית התואמת את המוסכמות שבשימוש. תרשים 1.2 מציג תגובה זו, כפי שהיא מוצגת בדפדפן Explorer.



הערה!

כמו בדרישת GET, השרת עשוי גם הוא להחזיר כמות נוספת של נתונים אל הדפדפן בצורת כותרות HTTP, אשר הסרנו אותם מדוגמה זו למען הבהירות. תוכל למצוא פרטים נוספים אודות כותרות HTTP בפרק 8.

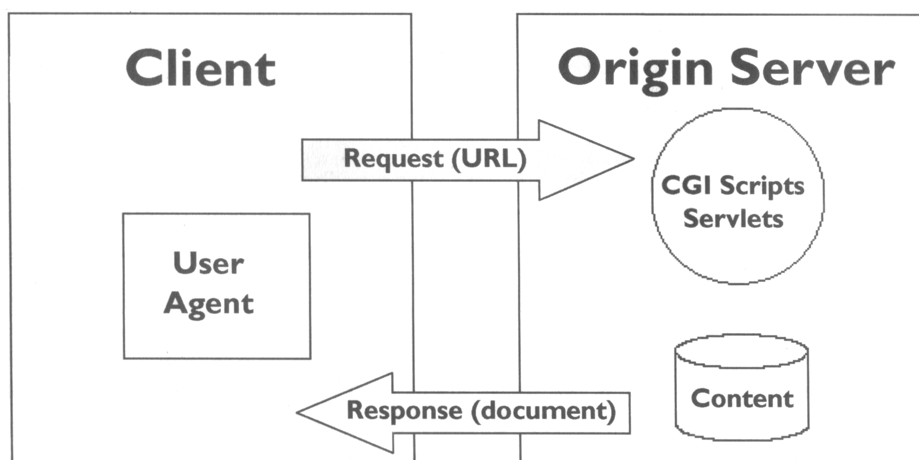
ה- Web היא מכלול פשוט. היא משתמשת במודל שידור פשוט של הודעות, או תנועות, שבו דפדפן Web (או תוכנה אחרת כלשהי) המשתמש ב-HTTP דורש קטע נתונים ייחודי וממתין לתגובה. תרשים 1.3 מסכם את מודל תכנות Web. כל הודעות HTTP הן הודעות טקסט שאינן מציגות את תבנית ההודעה הדחוסה ביותר.

הישות הדורשת נתונים משרת יכולה להיות כל סוג של תוכנה הפועלת על רוב סוגי החומרה. כל עוד ישות זו יכולה ליצור קשר TCP/IP ולהפעיל דרישת HTTP תקפה, היא יכולה להפעיל העברת נתונים ב-Web. המונח המקובל לישויות המסוגלות ליצור דרישות HTTP הוא **סוכן-משתמש** (user-agent).

בדומה, הישות הממוחשבת המגיבה לדרישת HTTP תקפה יכולה להיות כל אחת ממיגוון התוכנות הרבות הפועלות על סוגי חומרה שונים. כדי לשקף את האופי הכללי של תגובות, המונח המקובל לישות המגיבה לדרישות HTTP הוא **שרת-תוכן** (content-agent).



תרשים 1.2 התגובה ותבניתה



תרשים 1.3 מודל תכנות Web

הדברים אינם כה פשוטים כפי שהם נראים במבט ראשון. לדוגמה, יש פרוטוקולים נוספים מלבד HTTP ו-HTTPS, יש סוגים שונים של כתובות משאבים (URL) ויש מספר סוגים של דרישות שדפדפן יכול ליצור. הן סוכן-משתמש והן שרת-תוכן יכולים להכיל מידע רשות (אופציונלי) בכותרות הודעה מיוחדות. לקבלת מידע שלם על כך, עליך לקרוא את מפרט HTTP 1.1 הרשמי, RFC 2616.

סיבוכים נוספים יכולים להיות עקב הימצאות של **Gateway** אחד או יותר, או של **Proxy Servers**, בין סוכן-משתמש הדורש מידע לבין השרת המגיב לדרישה. Gateway מתפקד כמתווך לשרת אחר כלשהו. הדוגמה הנפוצה ביותר של Gateway היא שרת firewall ארגוני. הוא שולח דרישות מסוכני-משתמש שמחוץ לרשת פנימית משותפת, רק אם יש להם הרשאת גישה לרשת. Gateway יכול להשתמש ב-HTTP או באופן אחר כדי להתקשר עם שרת כלשהו כדי למלא משימה שקיבל. ייתכן שסוכן-משתמש אף לא ידע שהוא מתקשר עם Gateway.

כמו במקרה של Gateway, גם Proxy server משמש כמתווך בין סוכן-משתמש לבין שרת-תוכן. ה-Gateway פועל לטובת השרת, אך תפקיד ה- Proxy server להעביר או שלא להעביר דרישות של לקוחות שונים המופנות אל השרת. כלל זה תורם לכך ש-Proxy server חייב ליישם, בדרך כלל, הן את יכולות הדרישה והן את יכולות התגובה של HTTP. שרת כזה גם יכול לשמש כ**שרת יעד** (Destination server), אך במקרה זה הוא ממלא אחר דרישות של סוכן-משתמש.

אם אינך כותב תוכנה לניהול Gateway או Proxy server, אינך צריך לעסוק בשוני ביניהם. כל שעליך לדעת הוא שמחשב (מחשב Gateway) נמצא בין הסוכן-משתמש שלך לבין שרת-התוכן, ויש לו תפקידים מסוימים.



הערה!

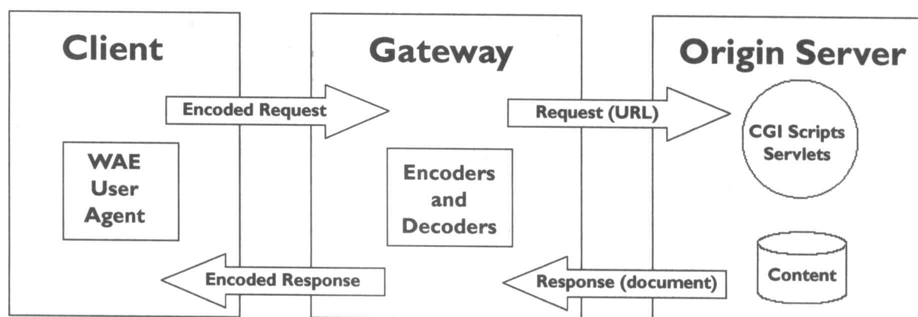
נהוג להשתמש במונחים Proxy ו- Gateway Interchangeably, וגם אנו משתמשים בהם בחופשיות במהלך הספר. בפרק זה אנו מעדיפים לציין Proxy Server, מפני שזהו המונח הנכון. בפרקים הבאים נעדיף לומר "WAP Gateway" מפני שהמונח פשוט וברור מאוד. אם תתבלבל, זכור ש- WAP Gateway הוא למעשה Proxy Server.

מודל התכנות WAP

מודל התכנות WAP המוצג בתרשים 1.4 דומה למודל התכנות Web, אלא שיש ביניהם שני הבדלים חשובים:

- תמיד קיים WAP proxy server אחד לפחות בין סוכן-משתמש לבין שרת-תוכן. התפקיד העיקרי של Proxy server זה הוא לתרגם פרוטוקולי WAP מסוכן-משתמש לתבנית HTTP, כדי לתקשר עם שרת-התוכן, וגם להיפך. עליו להדר לפעמים תוכניות WML ו-WMLScript דינמיות המגיעות משרת-התוכן לפני החזרתם אל סוכן-משתמש.
- התקשורת בין סוכן-משתמש לבין WAP proxy server נעשית באמצעות פרוטוקולי WAP. הפרוטוקול החשוב בהם הוא WSP (**W**ireless **S**ession **P**rotocol), שהוא למעשה צורה בינארית דחוסה של HTTP 1.1.

Proxy server משמש גם ככלי מפתח בניהול היבטים נוספים של העברת נתונים. לדוגמה, WAP proxy server אחראי לידיעת השפות וקבוצות התווים שכל התקן WAP מבין ולידיעת השפות המקובלות בשרתי התוכן. עליו לתווך בין השניים כדי שהמשתמש יקבל הודעה ברורה. WAP proxy server יכול לספק שירותי רשות (אופציונליים) נוספים שאינם כלולים במפרט WAP, כגון אחסון העדפות של מנויים וניהול דואר אלקטרוני.



תרשים 1.4 מודל תכנות WAP



הערה!

WAP proxy server ושרת-התוכן חייבים להיות מותקנים במחשבים נפרדים, או בנפרד באותו מחשב. שרת-תוכן יכול לכלול יכולות של WAP Proxy server. הדבר יכול להתאים לתצורות שבהן אתה חייב לספק אבטחת נתונים מקצה לקצה, ללא העלות של תקשורת HTTPS מלאה, עבור פתרונות וירטואליים מותאמים אישית לקהל נבחר, או עבור יישומים הדורשים זמן תגובה מקסימלי מובטח.

הבה נבחן את ההעברה שנידונה לעיל, ונביט במקבילה לה במערכת WAP.

המשתמשים פונים אל הכתובת www.worldfaq.com באמצעות הוספת URL זה להתקן WAP שלהם, ולחיצה על מקש מיוחד המסמן להתקן לפנות אל כתובת זו, או באמצעות הפעלת תוכנית היוצרת את הבקשה עבורם. כמו בדוגמה הקודמת, הסוכן-משתמש של WAP מפעיל בקשת GET עבור www.worldfaq.com:

```
GET/HTTP/1.1
```

הסוכן-משתמש משדר את בקשת GET אל שרת WAP, אך תחילה הוא משנה את ההודעה לתבנית בינארית דחוסה, המתאימה יותר לשידור בקשר סלולרי מאשר תבנית טקסט ASCII של HTTP. כאשר שרת WAP מקבל את ההודעה, הוא מנתח אותה ומאמת את הצורך לשלוח את ההודעה לשרת www.worldfaq.com. הוא ממיר את ההודעה ל-GET של HTTP מבוסס טקסט, ושולח אותה אל www.worldfaq.com. השרת www.worldfaq.com מקבל את הדרישה, מנתח אותה ושולח את תגובתו. במקרה זה, התגובה היא הודעת OK של HTTP בדומה לדוגמה הקודמת, אך כמסמך של שפת סימון סלולרית (WML) ולא כמסמך HTML, כמו גוף ההודעה exl-01.wml.

```
<?xml version="1.0"?>
<!DOCTYPE WML PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
  "http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
  <card>
    <p>
      WorldFAQ is a web site that provides
      information about places around the world.
      It is currently under construction.
    </p>
  </card>
</wml>
```

שרת WAP מקבל את התגובה, מתרגם אותה להודעה בינארית דחוסה ומשדר אותה להתקן הדורש. התקן WAP מקבל את התגובה, מנתח את גוף ההודעה ומציג את תוכן הקובץ index.wml על מסך הסוכן-משתמש של WAP. תרשים 1.5 מציג כיצד הדבר נראה בדפדפן Phone.com.

```
WorldFAQ is a web
site that provides
information about
places around the
world. It is
currently under
 OK
```

תרשים 1.5 התגובה בתבנית WAP

מודל התוכנה Web ומודל התוכנה WAP דומים בעיקרם, הודות למאמצי פורום WAP להשתמש בתקנים קיימים היכן שרק ניתן. דבר זה מקנה למפתחי יישומים יתרונות משמעותיים:

- מודל WAP פשוט וקל להבנה. תוכל להשתמש בידע הנוכחי שלך אודות פעולת Web כדי ליצור יישומי WAP חדשים.
- ביכולתך להשתמש ברבים מכלי פיתוח התוכנה הקיימים. למעשה, כדי ליצור תוכן דינמי תוכל להשתמש בכלי CGI הזמינים בשפות תוכנה שונות, להעברת מסמכי WAP אל סוכן-משתמש.
- סוכני-משתמש WAP פשוטים מאוד. כתוצאה מכך יישומי WAP פשוטים גם הם, כי הם מתמקדים בתצוגת התוכן ולא ביצירתו. ביכולתך לבודד את קטע הלוגיקה של היישום מתהליך העיבוד המסובך שבשרת. הפרדה זו מקלה על פיתוח ואחזקה של יישומים יותר ממה שמקובל במודולי תכנות מסורתיים.
- אם יש יישומים מבוססי Web קיימים, תוכל לשמר את ההשקעה שעשית במסדי נתונים, בתוכן קנייני, בלוגיקת יישום ובממשקי תכנות יישומים (API). למעשה, יש WAP proxy servers מסחריים זמינים, המסוגלים להמיר אוטומטית ישומים מבוססי Web ליישומי WAP כפעולה הנלווית להמרת תשובות HTML לתשובות WML. לעיתים, Proxy servers אלה יוצרים תוכן לא אופטימלי עבור טלפונים סלולרים, ולכן אינם יכולים להיות בחירה טובה.

ארכיטקטורת WAP

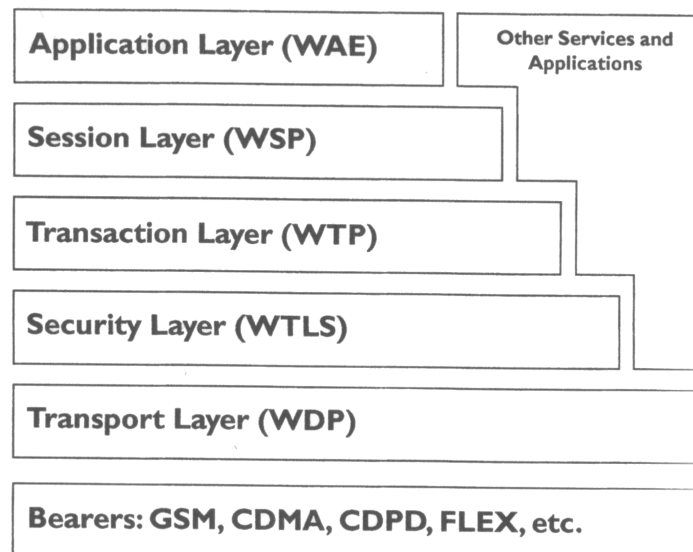
לאחר ההצגה הבסיסית של מודל תכנות WAP, הבה נתבונן בעיון בארכיטקטורת WAP עצמה, כפי שהיא מוצגת בתרשים 1.6.

ל-WAP יש ארכיטקטורת שכבות בדומה למודל הרשת OSI של ISO (**I**nternational **S**tandards **O**rganization). תוכל למצוא תיאורים מפורטים על כל שכבות ב-WAP. להלן סקירה קצרה בנושא.

כמפתחי יישומים אנו מתעניינים מאוד ב**סביבת יישומי WAP** – **WAE** (**W**AP **A**pplication **E**nvironment), שהיא השכבה העליונה של הארכיטקטורה. היא כוללת מפרטי התקן כלליים, שפות התכנות WML ו-WMLScript הדרושות לכתיבת יישומי WAP, ממשקי יישומים (API) של טלפוניה (WTA) לצורך גישת פונקציות טלפוניה מתוכניות WAE וגם קבוצה של תבניות תוכן מוגדרות, הכוללות גרפיקה, רשומות ספר טלפונים ומידע תאריך. בהמשך הפרק נדון בפירוט בסביבת היישום.

השכבה הבאה בארכיטקטורת WAP היא **פרוטוקול שיח סלולרי** – **WSP** (**W**ireless **S**ession **P**rotocol), המתייחס לשכבת ה- session. בפשטות אפשר לומר ש-WSP הוא בינארי, גירסה קולית של HTTP 1.1, שעוצב במיוחד עבור העברות תמונות דפדפן ברשתות סלולריות בעלות רוחב פס נמוך וזמן השהיה ארוך יחסית. בנוסף, פורום WAP הרחיק אל מעבר ליכולת הבסיסית של HTTP, והוסיף יכולות השהיה וחידוש מהירים של התקשורת.

חשוב! נוספה היכולת להעברת תוכן אמינה (מסירה מובטחת, Guaranteed delivery) שנוצרה על פי דרישה, והעברת תוכן שאינה אמינה (unreliable), המאפשרת לשרת לשלוח הודעות ומסמכים לסוכן-משתמש גם אם אינו נדרש במפורש לעשות זאת. בתיאור שלנו אודות מודולי תכנות Web ו-WAP ראינו שהסוכן-משתמש יוזם את הדרישה. לעומת זאת, יועיל מאוד אם שרת יודיע להתקן WAP על קריאת טלפון צפויה, דואר אלקטרוני (e-mail), פקס או הודעת איתורית. דחיפת התוכן מיועדת לאספקת סוגי שירותים אלה.



תרשים 1.6 ארכיטקטורת WAP

השכבה השלישית של ארכיטקטורת WAP היא **פרוטוקול WTP** (**W**ireless **T**ransaction **P**rotocol). זהו פרוטוקול העברה פשוט התומך בסוגי הודעות אלה: דרישות חד-צדדיות לא אמינות (דחיפה לא מובטחת), דרישות חד-צדדיות אמינות (דחיפה מובטחת) והעברות בקשה/דחיפה דו-צדדיות אמינות (דמויות דפדפן). הדבר הותאם באופן אופטימלי לרשתות סלולריות על ידי צמצום התקורה, כגון בדיקת רצף מנות שבשימוש TCP/IP.

WTLS (**W**ireless **T**ransport **L**ayer **S**ecurity) הוא פרוטוקול אבטחה המבוסס על התקן התעשייתי **TLS** (**T**ransport **L**ayer **S**ecurity), הידוע בשמו הקודם **SSL** (**S**ecure **S**ocket **L**ayer). פרוטוקול זה מספק שלמות נתונים, פרטיות, אימות ושירותי הגנה מסוג מניעת-שירות. כמפתח תוכנה, תוכל להפעיל את WTLS באופן זהה להפעלת דפדפן Web, תוך שימוש בפרוטוקול HTTPS במקום HTTP.

השכבה האחרונה של ארכיטקטורת WAP היא **פרוטוקול WDP** (**W**ireless **D**atagram **P**rotocol). שכבה זו מספקת ממשק יציב בין ספקי שירותים סלולרים רבים והשכבות היותר גבוהות של הארכיטקטורה. מתחת ל-WDP נמצאות מספר רשתות, ובכללן **CDPD** (**C**ellular **D**igital **P**acket **D**ata), **GSM** (**G**lobal **S**ystem for **M**obile Communication), **iDEN** (**I**ntegrated **D**igital **E**nhanced **N**etwork) ו-**CDMA** (**C**ode **D**ivision **M**ultiple **A**ccess).

WAE

סביבת יישומי WAP – **WAE** (**W**AP **A**pplication **E**nvironment) היא חלק מארכיטקטורת WAP הזמינה למפתחים של יישומי WAP. הסביבה מכילה מספר רכיבים:

- **מפרט מיקרו-דפדפן** המגדיר כיצד צריך לתרגם את WML ו-WMLScript בטלפון הסלולרי, וכיצד להציג את הנתונים למשתמש.
- **שפת סימון סלולרית** – **WML** (**W**ireless **M**arkup **L**anguage) המעוצבת לפיתוח יישומי WAP בלתי תלויים.
- **שפת תסריט WMLScript** להרחבת יכולות WML.
- **מסגרת ליישומי טלפוניה סלולרית** – **WTA** (**W**ireless **T**elephony **A**pplications), המספקת מנגנון שספקי שירות יכולים להשתמש בו ליישום פונקציות המאחדות את פונקציות הטלפון ופונקציות המיקרו-דפדפן של התקן WAP.

סביבת העבודה של היישומים הסלולרים (WAE) גם מניחה שקיים Proxy server זמין תואם WAP, המתרגם בין פרוטוקול WAP לפרוטוקולי תקשורת אינטרנט אחרים, כמו גם בין WAP לדרישות ותגובות בתבנית HTTP. כמו כן, השרת מספק שירותי מיטמון כדי לזרז את הגישה לתוכן. בנוסף לכך הוא מהדר תוכניות WML ו-WMLScript לערכים בינאריים תואמים, כדי להעבירן אל התקן תואם WAP והפעלתן על ידו. WAP proxy server עשוי לספק שירותים נוספים, כגון תרגום תבניות אינטרנט גרפיות לתבניות המובנות על ידי התקן WAP.



הערה!

רשת WAP המתוכננת כראוי חייבת לכלול Proxy server. השרת יכול להימצא במחשב נפרד, או כחלק משרת-תוכן.

Microbrowser

תיארנו את המאפיינים הבסיסיים של **מיקרו-דפדפן WAP** (WAP Microbrowser) – זהו תהליך המיישם את מודל התכנות WAP. הכוונה היא שכדוגמת דפדפן HTML, הוא מנהל את מנגנון הגשת הדרישות וקבלת התגובות וניתוחן, כמו גם את כל המשימות המשניות הקשורות לפעולה זו.

המיקרו-דפדפן כולל את המפרשים (Interpreters) של WML ו-WMLScript. הוא יודע כיצד לפרש את הקוד של שפות אלו, ולהחליט כיצד להציג את הנתונים בתצוגת ההתקן הסלולרי. כמו כן, דרושה לו ידיעה מפורטת ומדויקת של כתובות URL, מכיון שזו דרך הפעולה של יישומי WAP. לניהול העברת הנתונים המיקרו-דפדפן צריך לדעת גם כיצד לתקשר עם השכבות השונות של מחסנית פרוטוקול WAP כדי ליזום דרישה, להפעיל העברה מאובטחת, להשהות ולחדש פעילות אם דרוש, ועוד.

למיקרו-דפדפן WAP עשויות להיות יכולות נוספות, כמו לדוגמה מטמון (cache), בדומה לזה שבדפדפן HTML. במקרה זה עליו לדעת מה מצוי במטמון, מתי לאחזר URL מהמטמון ומתי להסיר ממנו פריט כלשהו. שלא כ-HTML, סביבת היישומים הסלולרים היא בעלת משתנים בעלי אורך חיים ארוך יותר מזה של מסמכים יחידים, ודבר זה מקל על פיתוח יישומים. המיקרו-דפדפן צריך לדעת את השמות ואת הערכים וכיצד לשלבם בביטויים כשנחוץ. למיקרו-דפדפן יכולה להיות מחסנית היסטוריה שתכיל את מספרי ה-URL בהם ביקר המשתמש לאחרונה.

המיקרו-דפדפן חייב להבין חלק מפרוטוקול HTTP 1.1, למרות שתפקידו של WAP Gateway לטפל בהעברת הנתונים בין פרוטוקול WAP לבין פרוטוקול HTTP. כאשר המיקרו-דפדפן שולח דרישה, עליו לדעת איזה כותרות לכלול בה, כך שתהיה מובנת ל- WAP proxy server ולשרת-התוכן. עליו לדעת גם כיצד לפרש את הכותרות הכלולות בכל תגובה.

לסיום, מיקרו-דפדפן WAP צריך לפעול בסביבת החומרה שאליה התכוון פורום WAP, שבה יש RAM ו-ROM מוגבלים בקיבולתם, מסכים קטנים, יכולות קלט/פלט מוגבלות ותקשורת רשת סלולרית. עבור דבר שאמור להתאים לסביבה כה מוגבלת, המיקרו-דפדפן חייב לדעת וגם לבצע דברים רבים.

WML

שפת סימון סלולרית – WML (**W**ireless **M**arkup **L**anguage), המוסברת בפירוט בפרק 4, מבוססת על תגיות כמו HTML ומעוצבת לאילוצי-חומרה, התקנים סלולרים בעלי רוחב פס צר ויכולות קלט/פלט מוגבלות. מסמכי WML משתמשים בדימוי של כרטיס וחפיסה (card-and-deck), שבה הכרטיס הוא יחידה אחת של אינטראקציית משתמש וחפיסה היא קבוצת כרטיסים משויכת. כדוגמת דף HTML, כרטיס טיפוסי מכיל תוכן נראה כלשהו, שניתן להוסיף בו מספר אפשרויות משתמש לבחירה, להוספת נתונים מסוימים, או לניווט לכרטיס אחר. ההוראות שבכרטיס עשויות להפעיל חפיסות סטטיות או דינמיות מתוך שרתי תוכן.

מפרטי WML מגדירים את התפקידים של תגיות ייחודיות, ולא כיצד סוכן-משתמש מסוים מיישם אותן. כפי שתראה בפרק 4, לכל ספק התקן WAP יש מידה גדולה של חופש להחליט כיצד להציג תגיות WML למשתמש, מהו מנגנון הוספת הנתונים הזמין, גודלו של המסך ועוד. WML היא שפה בלתי תלויה בממשק-המשתמש.

היכולות הכלליות של WML:

- **תמיכה בטקסט ובתמונות**, כולל עזרי תצוגה כגון קווי עצירה (line breaks), עיצוב וכדומה. התקני WAP אינם צריכים לתמוך בתצוגת תמונות.
- **תמיכה בקלט משתמש**, כולל הוספת טקסט, רשימות בחירה ופקדים המפעילים משימות. לדוגמה, ביכולתך להקצות URL ללחצן-התקן מסוים, כך שכאשר המשתמש לוחץ עליו, דרישת GET נשלחת לשרת URL הזה.
- **מיגוון מנגנוני ניווט**, המבוססים על סכמת השמות הסטנדרטית ל-URL שנקבעה באינטרנט, מאפשר לנוע בין כרטיסים שבחפיסה או בין חפיסות. כל התקן WAP יכול לשלב מנגנון היסטוריה עבור כרטיסים שכבר ביקרו בהם. כך, המשתמש יכול לבקר שוב בכרטיס קודם על ידי לחיצה על לחצן **Back**, בדומה ללחיצה על לחצן Back בדפדפן אינטרנט.
- **תמיכה בשפות שונות** על ידי שימוש בקבוצת התווים Unicode.
- **יכולות ניהול מצב והקשר**, ובעיקר יכולת העברת משתנים מחפיסה לחפיסה, יכולת החלפת משתנים ויכולת מיטמון משתנים וחפיסות על ידי הסוכן-משתמש, כדי למטב את ניצול המטמון ולמזער את הדרישות מהשרת.

WMLScript

WMLScript היא שפה פשוטה, אך רבת עוצמה, לכתיבת תסריטים. היא עוצבה כדי להרחיב את יכולות WML, בדומה ל-JavaScript שהרחיבה את יכולות מסמכי HTML. WML מטפלת בהליכי קלט/פלט, העברת תכנים וביצוע תהליכים (אירועים), אך היא חסרת יכולות חישוב משמעותיות. WMLScript ממלאת בהצלחה חלל זה, ומאפשרת להגדיר פונקציות הניתנות לקריאה מתוכניות WML. במסגרת הפונקציות העומדות

לרשותך ישנם גם משפטי ההתניה "if...then...else", משפטי השמה, קריאות לפונקציות, מבני לולאה, סוגי נתונים בסיסיים, כגון משתני Boolean או Integer ועוד. WMLScript כוללת קבוצת משימות שלמה – לוגיות ואריתמטיות, ואופרטורים להשוואה.

WMLScript היא שפה הניתנת להרחבה באמצעות ספריות. הגירסה WMLScript 1.1 עונה למפרטי WAP 1.1, וכוללת ספריות נקודה צפה, מחרוזות, URL, ספריות דו-שיח, ספריה לפונקציות שפה בסיסיות, כמו המרות סוג וניתוח מחרוזות וספריית פונקציות לפעולות ממשק עם מיקרו-דפדפן WAP.

למרות ש-WMLScript ניתנת להרחבה באמצעות ספריות חדשות, פורום WAP אינו מספק קבוצה פתוחה של ממשקי פיתוח יישומים (APIs) כדי שמפתחי תוכנה שונים יוכלו להגדיר ספריות WMLScript.

WTAI

מכיון שהתקני WAP הם ביסודם טלפונים, גם אם יותר חכמים מהטלפון הממוצע שבידך, הגיוני לשלב יכולות טלפוניה עם WAP. זוהי מטרתו של **ממשק יישומי טלפוניה סלולרית** – **WTAI** (**W**ireless **T**elephony **A**pplications **I**nterface), המהווה חלק ממפרט יישום טלפוניה סלולרית המפרט את ה-APIs. כמעט כל תכונות WTAI מעוצבות במיוחד עבור מפעילי רשתות סלולריות. הממשק מאפשר לפתח במהירות יכולות טלפוניה מתקדמות עם קבוצה מיוחדת של APIs, שניתן לגשת אליהם מ-WML ומ-WMLScript.

השימוש ב-WAE כדי ליצור יישומי טלפוניה הוא בעל יתרונות בולטים על השיטה הנוכחית של כתיבת יישומים ייחודיים להתקן, אשר נצרבים ל-ROM של המכשיר הידני. היתרון הבולט ביותר הוא יכולתו של ספק השירות לשדרג בקלות יישום WTAI על ידי שימוש במיקרו-דפדפן כדי להשיג את הגירסה האחרונה של התוכנית. כמו כן, הוא יכול לכתוב יישומי WTAI שיפעלו על כל ההתקנים תואמי WAP ורשתות תומכות WAP.

למרות שיישומי WTAI יכולים לנצל את מלוא היתרונות של מודל התכנות ממוקד-המסמך הנתמך על ידי WML ו-WMLScript, רוב ה-APIs של WTAI לספקי שירות משתמשים במודל אסינכרוני מונהג-אירועים, המדמה טוב יותר את הפעילות האמיתית של רשת טלפונים. לדוגמה, שיחה נכנסת היא אירוע המופעל מחוץ לתחום של התקן WAP. אינך יודע מתי הוא עשוי להתרחש. יישומי WATI צריכים להיות בעלי יכולת להגיב לסוג זה של אירועים.

מכיון שהממשק WTAI מיועד בעיקר לספקי שירות, נתעלם ממנו בהמשך הדיון בספר זה.



הערה!

אם ברשותך התקן WAP המשתמש באחת מפונקציות WTAI, יש לו בוודאי קשר ישיר לשרת WTA, שרת של ספק שירותים סלולרים שעוצב במיוחד לתמיכה בפונקציות טלפוניה. האינטראקציה בין ההתקן לבין שרת WTA היא בדרך כלל ישירה, ללא Proxy server באמצע הקשר. לעומת זאת, פונקציות WTAI עשויות להשתמש בשכבה אחת או יותר של מחסנית פרוטוקול WAP, כדי ליישם תקשורת עם השרת.

תבניות תוכן

כל תוכן המועבר מסוכן-משתמש אל Proxy server ואל שרת תוכן וחזרה, חייב להיות בתבניות ייחודיות תואמות את הפרוטוקול WAP. ראינו שתבנית ההודעה תואמת את מפרטי HTTP 1.1. תבניות HTTP נדחסות בעת העברתן מסוכן-משתמש אל Proxy server. ההודעות העוברות בין Proxy server לבין שרת תוכן הן תבניות HTTP סטנדרטיות מבוססות-טקסט.

מה לגבי גופי ההודעות המכילים את המידע הקריטי הנכלל בכל תגובה? כאשר שרת תוכן יוצר באופן דינמי מסמך WML, כדי לשלוח תשובה לסוכן-משתמש, הוא יוצר קוד מקור; זו בחירה גרועה עבור תוכן הנשלח בקשר סלולרי משום נפחו הגבוה. סביבת היישומים הסלולרים WAE כוללת מפרטים לקידוד בינארי של תוכניות WML ו-WMLScript. כל Proxy server אחראי לקבלת קוד המקור של WML ושל WMLScript והידורו לקוד בינארי המוחזר לסוכן-משתמש לביצוע. הסוכן-משתמש מכיל מפרשים הפועלים בקוד בתים (bytecode) עבור WML ו-WMLScript.

כל שרת-תוכן תואם WAP יכול להפעיל קוד WML ו-WMLScript באופן ישיר, לעקוף את שלב הפענוח, וכך להתעלם מה- Proxy server במהלך ההעברה. הבעיה בגישה זו טמונה בכך ששרת-התוכן צריך להכיל בנוסף גם את המפענחים הדרושים להבנת דרישות מקודדות של שכבת WSP, המגיעות מסוכני-משתמש. רוב מפתחי היישומים יעדיפו לא לעסוק בבעיות מהדרים וממירי פרוטוקול, אלא להתמקד ביישום שלהם בלבד.

בנוסף לפענוח WML ו-WMLScript, סביבת היישומים WAE תומכת גם בארבע תבניות תוכן: WBMP (**W**ireless **B**it**map**), MIME (Multipart Messages), vCard ו- vCalendar.

- **WBMP**. למרות שקיימות מספר תבניות גרפיות מקובלות לאינטרנט, הידועות והשימושיות ביותר הן GIF (**G**raphics **I**nterchange **F**ormat) ו-JPEG (**J**oint **P**hotographic **E**xperts **G**roup), אך אף לא אחת מהן מתאימה ממש לשידור ברשתות סלולריות. פורום WAP הגדיר עבור שידור סלולרי את התבנית WBMP (**W**ireless **B**it**map**). תבנית זו ניתנת להרחבה, ומתאימה באופן אופטימלי לסוכני-משתמש בעלי יכולת מחשוב מינימלית. למרות ש-WBMP היא למעשה התבנית הגרפית הרצויה עבור WAP, התקני WAP יכולים לתמוך גם בתבניות אינטרנט גרפיות מקובלות, כגון GIF.

- **MIME**. תבנית MIME נוצרה במקורה עבור דואר אלקטרוני, אולם הורחבה לקבוצת תבניות הודעה לשימוש כללי ומשמשת גם לדרישות HTTP ולהודעות תגובה (פנה ל- RFC 822 ול- RFC 2045 לדיון מפורט על תבניות MIME). ברמתה הפשוטה ביותר, הודעה זו היא אוסף של יחידות טקסט ASCII. במקרה שלנו, היא יכולה להיות מסמך WML, המוחזר משרת תוכן אל WAP proxy server כתגובה לדרישת מסמך. קצת יותר מורכב הוא המקרה שבו תוכנית ASCII מקודדת לקוד בתים של WML כדי לשדרה חזרה לסוכן-משתמש. MIME מטפלת בכך בקלות רבה.

  הדברים הופכים למעניינים כאשר ברצונך לשלוח הודעות מרובות כחלק מהודעה אחת גדולה, כפי שמקובל לרוב בסביבת העבודה הסלולרית. לדוגמה, נניח שאתה מבקש לקבל מסמך סטטי משרת תוכן. אם מסמך זה כולל התייחסות לקובץ גרפי, המסמך נקרא ומנותח תחילה, ואחר כך נקרא הקובץ הגרפי בפעולה נפרדת. ברשת סלולרית פעולה זו יכולה להוסיף תקורה רבה להעברה פשוטה. אם ידוע לך בוודאות שאתה זקוק לקובץ הגרפי, מדוע לא לשלוח אותו יחד עם המסמך המקורי? הדבר עשוי לחסוך את הייזום וההמתנה של ההעברה השנייה. ההודעות המרובות של MIME מאפשרות לארוז מסמכים מרובים (או גרפיקות, קבצים בינאריים, או כל דבר אחר ש-MIME מבינה) למסמך יחיד.

- **vCard ו-vCalendar.** תבניות vCard ו-vCalendar הן תקנים בינלאומיים המגדירים תבניות של כרטיסי ביקור והודעות פגישה. למרות שהם מוגדרים כחלק של מפרטי WAP, נכון להיום הם אינם נתמכים על ידי רוב ספקי WAP. לא נדון בהם יותר בספר זה.

סוכני-משתמש (User Agents)

עד כה התייחסנו להתקן WAP ולכל התוכנה שבו, הדורשת מידע משרתי תוכן, בכינוי הכללי **סוכן-משתמש** (user agent). מונח זה מוגדר בבירור במפרט HTTP 1.1. אחד ממפרטי המפתח של WAP 1.1 מגדיר סוכן-משתמש של HTTP 1.1 כ-"לקוח המייצר דרישה. לקוחות אלה הם לעיתים דפדפנים, עורכים, spiders (Web-traversing robots), או כלים אחרים של משתמש קצה".

מפרט WAE מבהיר בבהירות שהוא מגדיר מסגרת כללית בלבד, ולא קבוצה קשוחה של מפרטים. תוצאתה של גישת מסגרת זו נובעת במידה מסוימת מהמשמעות הבלתי ברורה של המונח **סוכן-משתמש**. בתחילה, מפרטי WAP הקנו למונח זה את המשמעות שהיתה מקובלת במפרטי HTTP 1.1 – הלקוח אשר יוזם דרישה למידע והמשתמע מכך הוא שהדפדפן הוא סוכן-משתמש.

במפרט WAP מונח זה מוגדר באופן מפורש יותר. יש סוכן-משתמש המציג מסמכי WML או כזה המפעיל תוכניות WMLScript. יכול להיות גם סוכן-משתמש המספק תמיכה ביישומי טלפוניה סלולרית. להבחנה זו אין קשר ליכולת ליצור דרישה. למעשה, עוסקים כאן בסוג של שירות.

מפרט WAP מציין גם שאין כלל ביטחון שסוכן-משתמש כלשהו שאתה עשוי למצוא בהתקן WAP ידני נמצא בשליטת ההתקן. ייתכן שיהיו תהליכים אחרים המנהלים את התיאום הכולל, כאשר התקן WAP מופעל. עשויים להיות יישומים נוספים, כגון עורכי הודעה או ספרי טלפון, הדורשים משאבי חומרה. כל שהמפרט מנסה לעשות הוא להגדיר את התנהגות סוכני-משתמש WAP – שהם ביסודם שירותים ותבניות הדרושים לצורך הבטחת ההידודיות (interoperability) בין יישומים. גישה זאת טוענת שסוכן-משתמש הוא מטלת ביצוע שיכולה לפעול בעצמה ונבדלת מסוכני-משתמש אחרים.

אין כל חשיבות כיצד אתה מפרש את המינוח. השאלה החשובה היא מה אתה, כמפתח יישומים, מצפה למצוא בהתקן WAP, ותוכל לנצל בעת בניית יישומים? בשלב זה, פורום WAP עדיין נאבק בשאלה מהם הרכיבים ההכרחיים ומהם רכיבי הרשות של מפרט WAE. נראה שהפורום ידרוש שהמרכיבים הבאים יהיו בכל התקני WAP:

- מפרשי WML ו-WMLScript.
- תמיכה בגרפיקת WBMP, אם ההתקן תומך בגרפיקה.
- הפונקציות הציבוריות WTAI.
- מחסנית פרוטוקול WAP שלמה, מלבד Wireless Transport Layer Security (העברה סלולרית מאובטחת, שהינה שכבת ההעברה הסלולרית), הנדרשת בהתקנים התומכים בהבטחת שידור בלבד.

המנוע המפעיל את כל היכולות האלו, ובמובן מסוים קושר אותן יחד על ידי הספקת שירותים משותפים, הוא המיקרו-דפדפן.

בהמשך הדיון בספר זה נשתמש במונח סוכן-משתמש עבור מכלול סוכני-המשתמש הזמינים בהתקני WAP. היכן שדרוש, נשתמש במונח זה כדי לזהות שירותי WAE ייחודיים, כדוגמת סוכן-משתמש של WML. זהו אמנם מינוח מעורפל, אך אין הוא צריך להפריע בהבנת WAP.

בלי להתחשב בשמות רכיבי WAP השונים – סוכני-משתמש, שירותים, או מטלות ביצוע – הם חייבים לפעול באופן הידודי (אינטראקטיבי) זה עם זה. הנקודות הבאות מסכמות כיצד הדבר נעשה:

- תוכל ליצור קישור לפונקציות WMLScript ממסמכי WML על ידי שימוש ב-URL. שם מסמך URL מציין את המיקום בספריית WMLScript; חלק של שם ה-URL מציין את שם הפונקציה שבתוך הספריה.
- לא ניתן לקרוא למסמכי WML ישירות מפונקציות WMLScript, אולם ביכולתך לגשת למשתני מיקרו-דפדפן, שכתוצאה מכך יכולים להשפיע על ביצועי WML. ביכולתך גם לציין בפונקציה WMLScript מהו כרטיס WML שברצונך להגיע אליו לאחר סיום פעולת הפונקציה.

- תוכל לקרוא לפונקציות ציבוריות של WTAI הן ממסמכי WML והן מפונקציות WMLScript. מ-WML עליך להשתמש ב-URL מסוים, המזהה את פונקציית WTAI שברצונך לקרוא. מתוכנית WMLScript, תוכל לגשת לפונקציה WTAI ציבורית על ידי קריאה לה.

בניית יישומי WAP

בפרק 4 נעבור לדון בפרטי הבנייה של יישומי WAP בכלים של WML. לפני שנעשה זאת, אנו רוצים להכין אותך לקראת תכנות, בדיקה וניפוי של קוד WAP.

הנה מה שדרוש לך:

- **ערכת פיתוח תוכנה תואמת WAP 1.1 (SDK)**. יש ערכות SDK מסוגים שונים ורבות מהן ניתנות חינם. בדוק את אתר Web של פורום WAP, כדי ללמוד מהי הרשימה העדכנית. ערכות אלו פועלות בדרך כלל במערכות Windows 95/98 או בגירסה כלשהי של Unix או Linux. אנו השתמשנו בערכה של Phone.com.
- **שרת Web.** שרת זה יכול להימצא במחשב שערכת SDK נמצאת בו, או שהוא יכול להיות נפרד, ושניתן לגשת אליו מ-SDK.

עליך להגדיר את השרת באופן שיזהה כראוי ויוכל לשרת סוגי WAP MIME תקפים. נכון לעכשיו אלה הם:

Description of type תיאור הסוג	Associated extension סיומת משויכת	Content Type (MIME) סוג תוכן (MIME)
WML source code	wml	text/vnd.wap.wml
WMLScript source code	wmls	text/vnd.wap.wmlscript
Wireless bitmap	wbmp	image/vnd.wap.wbmp
Compiled WML	wmlc	application/vnd.wap.wmlc
Compiled WMLScript	wmlsc	application/vnd.wap.wmlscriptc

רוב ערכות SDK משתמשות בסימולטור הנראה ופועל כהתקן WAP טיפוסי, עם יכולת הפעלה של WMLScript ו-WML. באמצעות עורך טקסט סטנדרטי אפשר ליצור קבצי מקור של WMLScript ו-WML, ואחר כך יש להעביר אותם למקומות המתאימים בשרת Web שלך. בשלב זה תוכל להפעיל תוכנית WML משורת פקודה בסימולטור. התוכנית נטענת לסימולטור, מנותחת ונבדקת, כדי לאתר בה שגיאות. אם אין שגיאות, התוכנית נטענת על ידי מפרש קוד הבתים (bytecode) של הסימולטור ומופעלת.

רוב ערכות SDK מספקות מספר תכונות נוספות, כגון כלים לבדיקת תכולת המטמון, קוד המקור של החפיסה הנוכחית המופעלת, שמות משתנים, מחסנית היסטוריה ועוד. בשלב זה, WAP הינו טכנולוגיה חדשה ולכן, כלי הפיתוח עונים על צרכים בסיסיים, הם ילכו ויתפתחו ככל שיצטרפו מפתחי יישומים נוספים לשוק.

כשתגיע לרגע שבו תרצה ליצור ולבדוק תוכן Web דינמי, תצטרך להחליט כיצד ברצונך ליצור תוכן זה. רוב שרתי Web תומכים בתסריטי Perl ובתוכניות C להפעלת תוכן CGI. אחרים תומכים ב- Java servlets, ASP ומנגנונים אחרים. בדוק את התיעוד שלך כדי לדעת מה זמין, וכיצד עליך להגדיר אותו. בדרך כלל הדבר מחייב הגדרת דגלי תצורה מסוימים כדי להפעיל את יכולות CGI, ולהורות לשרת כיצד לזהות את תוכניות CGI כאשר הן נרשמות ב-URL. הדבר נעשה בדרך כלל על ידי סיומות קובץ או מיקום בתוך תיקיית השרת.

משהתחלת להפעיל תוכן דינמי, ייתכן שתרצה להתעמק בפרטי מחזור פעולת דרישה/תגובה של WAP ברמתה הנמוכה. כדי לבצע זאת, עליך להיות מסוגל לבחון את כותרות HTTP הנוצרות על ידי הסימולטורים של SDK ונכללות בדרישות הנשלחות על ידם אל השרת, וגם לבחון את כותרות התגובה הנשלחות חזרה מהשרת.

כדי לבצע את מה שנאמר לעיל, עליך לאפשר כניסה (logging) לשרת Web, כדי שיפיק את כל הכותרות בסוג כלשהו של קובץ יומן. לא כל שרתי Web יכולים לספק רמה זו של פירוט. אם זאת בעיה, תוכל ודאי להשיג בעצמך העתקים של כותרות אלו מתוך תוכניות CGI שלך, ולרשום אותן בקובץ היומן.

יש ערכות SDK המאפשרות לראות את כל מה שחוזר מהשרת אל הסימולטור, אחרות מסתירות רמה זו של פרטים. אם הערכה שבשימושך אינה מאפשרת לראות את כותרות התגובה, השתמש ב-Telnet.

לבסוף, עליך לבחון את היישום שלך בעולם האמיתי. כדי לעשות זאת, דרוש התקן תואם WAP, כיסוי רשת סלולרית עבור התקן זה, דרך בה יוכל ההתקן לפנות ל-WAP Proxy server, ודרך בה השרת יוכל למצוא את היישום שלך. לרבים מספקי WAP יש Gateways שתוכל להשתמש בהם אם תצטרף לתוכנית הפיתוח שלהם. ייתכן שתצטרך אתר Web פומבי גלוי שה-Gateway יוכל לגשת אליו.

כעת, משהבנת במה עוסק WAP, הבה נעבור לפיתוח יישומי WAP.

פרק 2

כלים והגדרה

פרק זה נכתב על ידי שרון טל

פיתוח יישומי WAP אינו מחייב סביבת עבודה מוגדרת. ניתן לכתוב מסמכי WML ו- WMLScript בעורכי טקסט רגילים כדוגמת Notepad ו- WordPad, אם כי אפשרות זו הופכת את משימת הפיתוח לקשה ובלתי יעילה. בעיה נוספת הטמונה בפיתוח באמצעות עורכי הטקסט היא חוסר היכולת לבדוק את תקינות המסמכים באופן מקומי.

כמענה לצורך זה פיתחו חברות התקשורת המובילות ערכות פיתוח (SDK) המסייעות בכתיבת יישומי WAP, ומאפשרות את בדיקת תקינות היישום ואופן הצגתו על המסך הסלולרי.

ערכות הפיתוח ניתנות להורדה מאתרי האינטרנט של יצרניות הטלפונים הסלולרים. הערכות הנפוצות כיום בקרב מפתחי ה-WAP הן ערכות הפיתוח של Nokia, Ericsson ושל phone.com.

הערכה המומלצת היא הערכה של Nokia, אשר נבדקה ונוסתה בהצלחה במשרדנו, ואותה תוכל להוריד מהאינטרנט, מהכתובת: http://www.nokia.com. הערכה נקראת Nokia WAP Toolkit.

לשם הפעלת ערכת הפיתוח של Nokia עליך להוריד מהאינטרנט ולהתקין את ערכת הפיתוח (SDK) של SUN, מהכתובת http://java.sun.com.

ערכות הפיתוח מורכבות משני חלקים עיקריים: emulator, המדמה כמעט באופן מושלם את פעולת המכשיר הסלולרי, וסביבת הפיתוח בה נכתב הקוד.

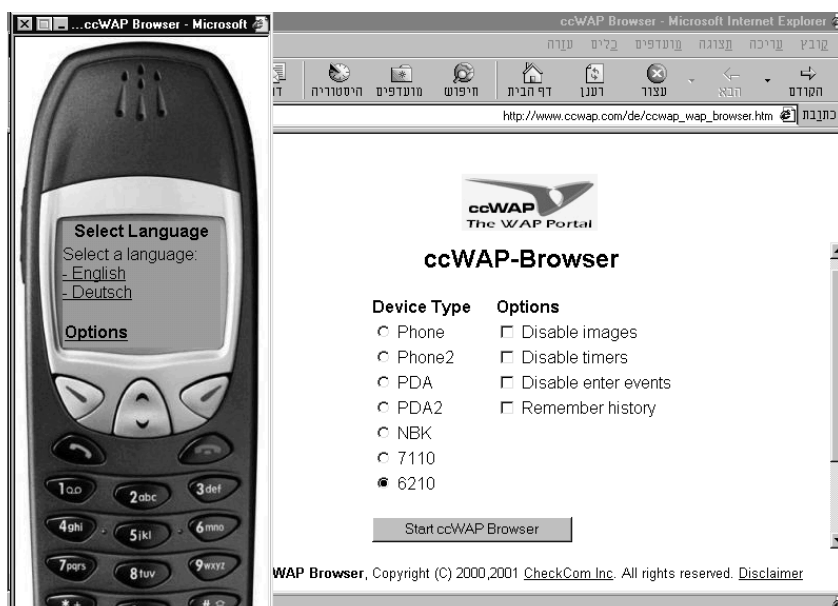
תרשים 2.1 מציג את ערכות הפיתוח של Nokia בפעולה.



תרשים 2.1

ערכות הפיתוח מאפשרות לבדוק את תקינות הקוד ולהציג את התוצאה על ה- emulator.

חשוב ביותר! אפליקציות ה- WAP אינן מתנהגות באופן זהה בכל המכשירים. לכן יש חשיבות רבה לבדיקת הקוד מול כמה מכשירים כדי להגיע לאופטימיזציה. את זאת תוכל לבצע באמצעות אתר http://www.ccwap.com. באתר זה תוכל לבחור את סוג המכשיר עליו אתה מעוניין לבדוק את הקוד. בדיקת הקוד מתבצעת On line ללא צורך בהורדת תוכנה כלשהי על ידי הזנת ה-IP של המחשב בו נמצאים הקבצים (ראה תרשים 2.2).



תרשים 2.2

יצירת תמונות בפורמט WBMP

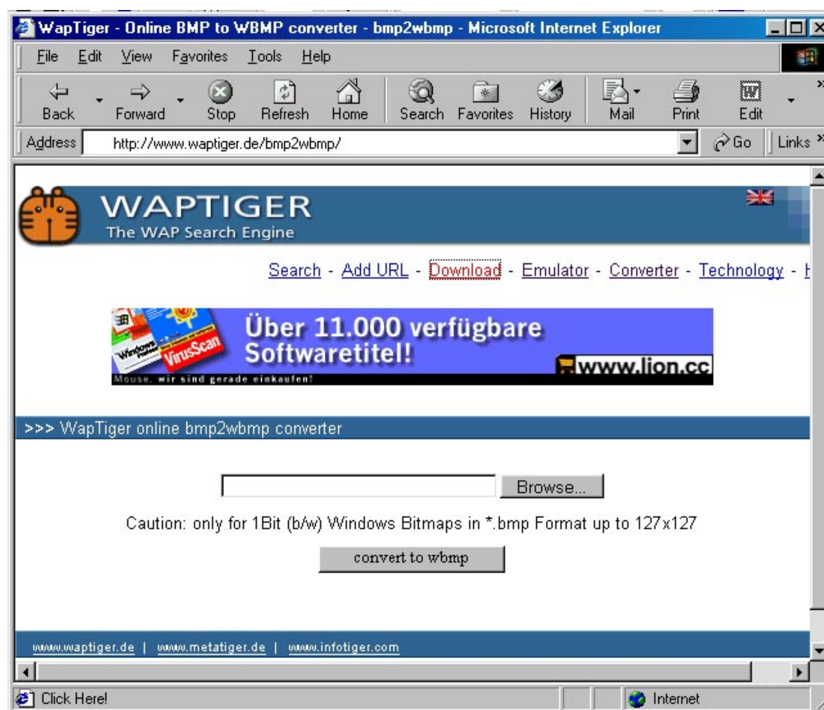
כאמור, תבנית התמונה היחידה הנתמכת כיום היא WBMP. כיצד אם כן יוצרים תמונות בפורמט WBMP?

בשלב ראשון יש ליצור באמצעות תוכנה גרפית תמונה בפורמט bmp ‏(1 bit) שחור-לבן.

לאחר ששמרנו את התמונה ניתן להמיר אותה לפורמט wbmp באמצעות תוכנות המרה ייעודיות.

ניתן לבצע את המרת התמונה online באתר http://www.waptiger.de/bmp2wbmp

תרשים 2.3 מציג את דף הפתיחה של יישום ההמרה.



תרשים 2.3

תהליך ההמרה אורך שניות ספורות, כאשר בתום התהליך ניתן להוריד למחשב את הקובץ בפורמט wbmp.

פרק 3

הגדרת PWS כשרת WAP

פרק זה נכתב על ידי שרון טל

הפעלת דפי WML חייבת להתבצע באמצעות שרת Web. בפרק זה נלמד כיצד להגדיר את ה- Personal Web Server כשרת WAP, ולהציג באמצעותו את דפי ה- WML על גבי המכשיר הסלולרי עצמו.

בטרם נוכל לצפות בפרי מלאכתנו על הצג הסלולרי, עלינו לבצע את הפעולות המפורטות בסעיפים שלהלן.

התקנת Personal Web Server

התקנת Personal Web Server במחשב בו מותקנת מערכת ההפעלה Windows 95 דורשת לבצע עדכון ל- Winsock 2.0. עליך להתקין את קובץ העדכון W95ws2setup.exe שבתיקיה software\Upgrade\ בתקליטור המצורף לספר זה. לאחר ביצוע העדכון יש לאתחל את המחשב. לאחר שהמחשב "עלה" מחדש:

1. הפעל את הקובץ **Setup.exe** שבתיקיה **Software\PWS** בתקליטור המצורף לספר.

   לידיעתך: במחשב בו פועלת מערכת ההפעלה Windows 98, הפעל את קובץ ההתקנה של Personal Web Server בתיקיה X:\add-ons\pws שבתקליטור ההתקנה המקורי של Windows 98 (החלף את האות X באות המייצגת את כונן התקליטורים שלך).

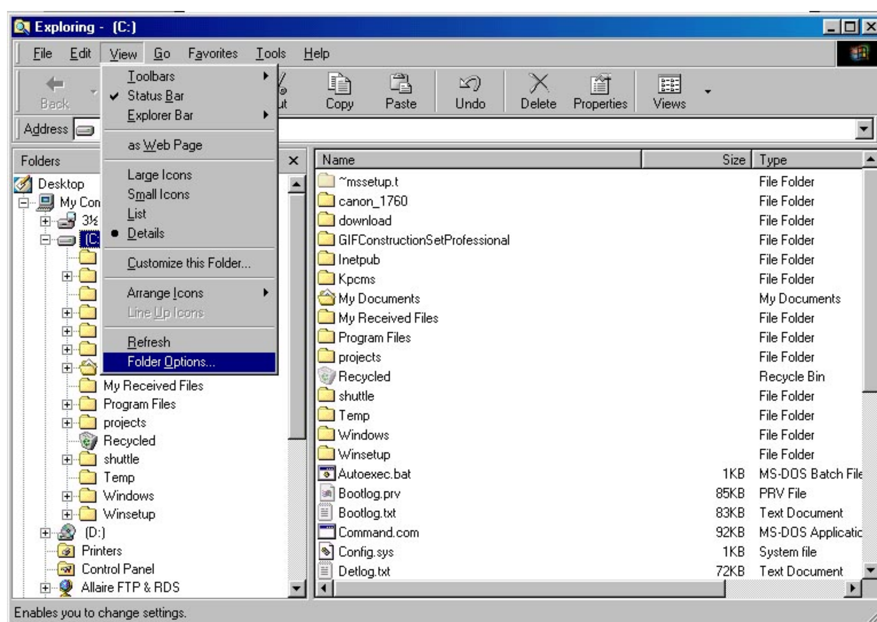
2. בחלון הפתיחה לחץ על **הבא** (Next).
3. בחלון הסכם השימוש בתוכנה לחץ על **Accept**.
4. בחלון סוג ההתקנה לחץ על **Typical**.
5. בחלון התקנת תיקיית ברירת המחדל לדף הבית לחץ על **הבא** (Next).
6. לסיום ההתקנה יש ללחוץ על **סיום** (Finish) ולאתחל את המחשב.

הגדרת MIME TYPE

כדי לאפשר ל-PWS לזהות סוגי WAP MIME תקפים, יש להגדיר את ה- MIME TYPE הבאים:

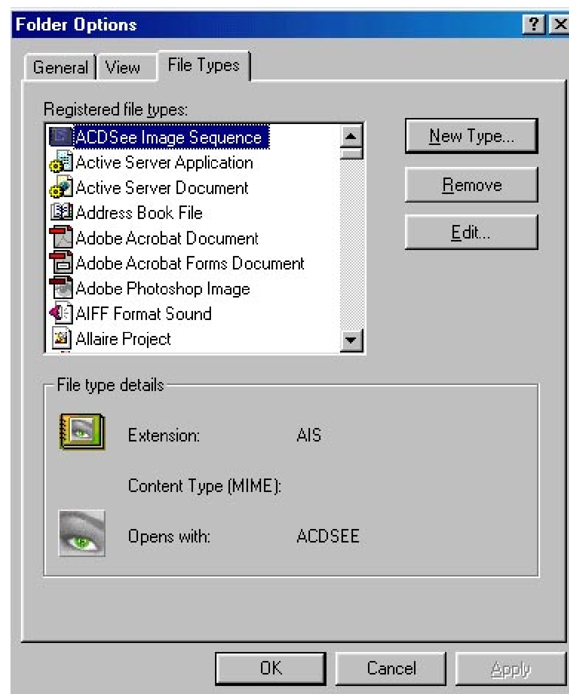
Description of type תיאור הסוג	Associated extension סיומת משויכת	Content Type (MIME) סוג תוכן (MIME)
WML source code	wml	text/vnd.wap.wml
WMLScript source code	wmls	text/vnd.wap.wmlscript
Wireless bitmap	wbmp	image/vnd.wap.wbmp
Compiled WML	wmlc	application/vnd.wap.wmlc
Compiled WMLScript	wmlsc	application/vnd.wap.wmlscriptc

1. פתח את **סייר Windows** (Windows Explorer), פתח את תפריט **תצוגה** (View), ובחר ב**אפשרויות תיקיה** (Folder Options).



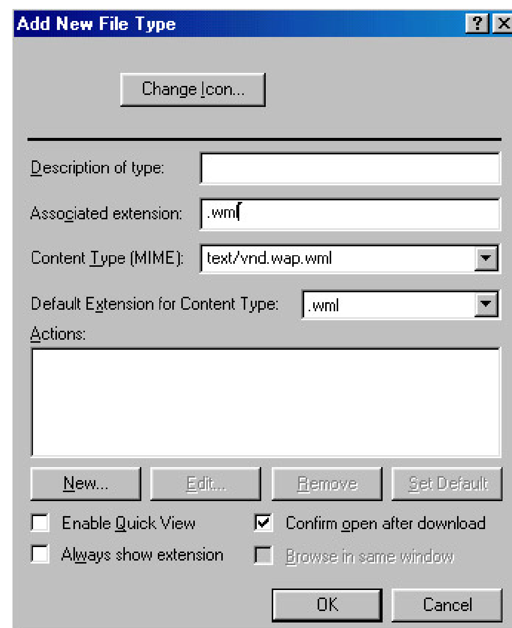
תרשים 3.1

2. בתיבת הדו-שיח **אפשרויות תיקיה** (Folder Options) בחר בכרטיסיה **סוגי קבצים** (File Types) (תרשים 3.2).



תרשים 3.2

3. בחר באפשרות **סוג חדש** (New Type).
4. בשדה **Associated extension** הקלד **.wml**
5. בשדה **Content Type (MIME)** הקלד **text/vnd.wap.wml**
6. לחץ **אישור** (OK).
7. חזור על סעיפים 3 עד 6 עבור שאר השורות שבטבלה הגדרות MIME Type.
8. בסיום הגדרת ה- MIME TYPE אתחל את המחשב.



תרשים 3.3

כעת מוכן ה- Personal Web Server להצגת דפי WML על גבי צג המכשיר הסלולרי.

פתח עורך טקסט, וכתוב את הקוד הבא:

```
<?xml version="1.0" encoding="ISO-8859-8" ?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
  <card id="first" title="my first wml page">
     <p>hello world</p>
  </card>
</wml>
```

שמור את הקובץ עם סיומת wml בתיקיה wwwroot.

הזן את כתובת ה-IP של המחשב שלך (תוך ציון שם הקובץ שיצרת) לשדה ה-URL במכשיר הסלולרי שברשותך, ואתה באוויר.


http://213.8.6.15/first.wml
ERICSSON
my first wml page
hello world
YES
NO
R520m

תרשים 3.4

פרק 4

WML

WML – **W**ireless **M**arkup **L**anguage (שפת סימון סלולרית) גירסה 1.1 עוצבה לצורך יצירת יישומים הפועלים על התקנים ניידים קטנים, כגון טלפונים סלולרים ועזרים דיגיטליים אישיים אחרים. יסודותיה של WML מקורם בתקנים שונים של האינטרנט. כתוצאה מכך, WML דומה במידה מסוימת ל-HTML4, שהיא השפה העיקרית להעברת נתונים ב-Web (או WWW - **W**orld **W**ide **W**eb). HTML עצמה מבוססת על **S**tandardized **G**eneral **M**arkup **L**anguage (SGML) שבתקן ISO-8879.

יסודות WML

WML ירשה את רוב המבנים התחביריים שלה מ-XML (E**x**tensible **M**arkup **L**anguage), שהיא תת-קבוצה מוגבלת של SGML, ומתקן World Wide Web לשפות סימון לאינטרנט. WML מוגדרת כיישום של XML.

אלמנטים

אם השפות HTML, SGML או XML מוכרות לך, הרי ש-WML לא תהיה חידוש של ממש. WML מוגדרת על ידי קבוצת אלמנטים, שלכל אחד מהם יש תגית ייחודית המוצגת באותיות רגילות. יש שתי דרכים להציג זאת. הדרך האחת היא:

```
<tag>
 .
 . tag contents
 .
</tag>
```

הדרך השנייה היא:

```
<tag/>
```

כך נוהגים כאשר האלמנט איננו יכול להכיל כל תוכן ויזואלי, כאשר אלמנט שיכול להכיל תוכן הוא ריק, או כאשר יש רק תגית פתיחה, ולא תגית סגירה (<go href="errcard" />, לדוגמה). לדוגמה, תגית הפסקת שורה אינה יכולה להכיל כל תוכן, ולכן תבניתה היא:

מכיון שכל אלמנט WML הוא בעל שם תגית ייחודי, נהוג להשתמש במונחי האלמנט והתגית לחילופין.

Properties

לכל אלמנט WML יש תכונות (properties) שמתארות היבטים שונים שלו, או שאין לו כלל תכונות. תכונות מוכרזות על ידי הוספת רשימת מילות-מפתח באותיות רגילות והגדרותיהן, בין גרשיים יחידים או כפולים לתגית התחילית. הזוגות ערך/תכונה מופיעות לאחר התגית הפותחת.

לדוגמה, לאלמנט של פיסקה יש שתי תכונות רשות, align (יישור) ו- mode (צורה). הן מתארות את יישור הטקסט ומאפייני גלישת המילים של הטקסט שבא לאחר תגיות הפיסקה <p>. להלן דוגמה של אלמנט פיסקה, המגדיר יישור לשמאל של טקסט הגולש משורה אחת לבאה אחריה.

```
<p
 align="left"
 mode="wrap"
>
.
. paragraph content
.
</p>
```

קיימת תכונה אחת הניתנת לשימוש עם כל האלמנטים של WML המכילים תוכן שניתן להצגה: xml:lang. תכונה זו מגדירה את סוג השפה הרשמית, או את הניב שווה הערך, שתוכן האלמנט משתמש בהם. כך ניתן לסוכן-משתמש של WML רמז המסייע לו להציג כראוי את התוכן. תוכל למצוא תגיות תקפות ב- RFC 1766.

יש שתי תכונות רשות נוספות הניתנות לשימוש עם אלמנטים של WML: id ו-class. id (זיהוי) של אלמנט הוא מציין ייחודי של אותו אלמנט הנמצא בחפיסה. זו היחידה הקטנה ביותר של WML הניתנת לשידור להתקן WAP. התכונה class (מחלקה, או סוג) מצרפת אלמנט עם שם מחלקה אחד או יותר, באמצעות שימוש בזיהוי שלו. כל האלמנטים בחפיסה ששם התכונה class שלהם זהה, נחשבים חברים (members) באותה מחלקה.

התכונה id בתגית <card>, והתכונה class משמשות בעיקר לפעולות צד-השרת, כגון שינוי סגנון של גיליון, אך כיום סוכן-משתמש WAP מתעלם מהן. בסופו של דבר, סוכני-משתמש WAP יתמכו בוודאי ב-DOM (**D**ocument **O**bject **M**odel) ותכונות אלו דרושות כדי לתמוך ב-DOM.

בהמשך ספר זה, כאשר מוגדרות תכונות של אלמנטים מסוימים, לא כללנו בדרך כלל את התכונות class, id ואת wml:lang לצורך פשטות הצגת הדברים. תמיד עליך להניח שאלמנט יכול לכלול את התכונות id ו-class, וכל אלמנט שיכול להכיל תוכן – יכול לכלול גם את התכונה xml:lang.

תצוגת תחביר

עד סוף הפרק אנו מתארים בצורה לא פורמלית את האלמנטים של WML ואת תכונותיהם. תחילה נציג אלמנט בדרך הדומה לזו של אלמנט הפיסקה שזה עתה תיארנו. נציין את שם האלמנט, בעקבותיו – את רשימת התכונות שלו, ואחריה את רשימת הפריטים שיכולים להיכלל כחלק מתוכן האלמנט:

```
<element
    רשימת תכונות וערכיהן
>
    דברים היכולים להיכלל בתוך האלמנט
</element>
```

רשימת התכונות כוללת את כל התכונות האפשריות, מלבד xml:lang, class ו-id (id נכללת בכרטיסים) וערכיהן. אם תכונה יכולה לקחת ערך מרשימת ערכים, אנו מציינים את האפשרויות באותה שורה, כשהן מופרדות בקו אנכי. ערך ברירת המחדל של תכונה מופיע בהדגשה. תכונות נדרשות מודגשות גם הן. להלן תחביר מלא של אלמנט פיסקה:

```
<p
    align="left | center | right"
    mode="wrap | nowrap "
>
    <a>, <anchor>, <br/>, DENTRY, <do>, FMTTEXT, <img/>, <table>, TEXT
</p>
```

יש סוגים שונים של ישויות שיכולות להיכלל בתוכן של אלמנט. הסוגים היותר נפוצים הם אלמנטים אחרים. לדוגמה, כמתואר בדוגמה הקודמת, תוכל לכלול תמונות **<img/>**, כתובות URL (**<a>**,**<anchor>**) וטבלאות (**<table>**) בפיסקה.

יצרנו קבוצה של קיצורים (כגון FMTTEXT) המפורטים בטבלה 4.1, כדי להציג את קבוצות האלמנטים הנדרשים ביותר ופריטים אחרים שעשויים להיכלל באלמנט או תכונה. הקיצורים כתובים באותיות רישיות וכך קל לזהותם. קיצורים אלה ודאי אינם מובנים כעת, אלא אם כן יש לך ניסיון מסוים ב-WML. ככל שתתקדם בקריאה בספר תכיר אותם יותר.

בנספחים לספר זה כללנו כמה נתוני תחביר של WML:

נספח א' הוא רשימה אלפביתית של כל האלמנטים של WML, התכונות שלהם והפריטים שביכולתם להכיל כתוכן. הנספח הוא סיכום של האלמנטים הנפרדים המתוארים בפרק זה.

נספח ב' הוא טבלת Cross Reference, המציגה את האלמנטים שיכולים להיכלל באלמנטים של WML. היא עוצבה לצורך קבלת מידע במהירות. לדוגמה, האם אתה יכול להכיל את האלמנט <**refresh**> בפיסקה.

נספח ג' הוא טבלת Cross Reference, המראה את התכונות שבשימוש כל אלמנט. היא עוצבה לצורך קבלת מידע מהיר. לדוגמה, מהו האלמנט שעליך להשתמש בו כדי להגדיר יישור טקסט.

למרות שהושקע זמן רב בהכנת נספחים אלה כדי להקל עליך בהבנת הדקויות של WML, קיים רק מקור סמכות אחד: ה- **D**efinition **T**ype **D**ocument WML (DTD). מקור זה מהווה את התיאור הפורמלי של WML. תוכל למצוא אותו בכתובת www.wapforum.org/DTD/wml_1.1.xml. כאשר אתה מתלבט, פנה למקור זה.

טבלה 4.1 קיצורי תוכן WML

DENTRY	אלמנטים של הוספת נתונים: **<input>**, **<select>** ו- **<fieldset>**
DOCHOICES	אפשרויות <**do**> תקפות: **delete**, **option**, **help**, **prev**, **accept** ו- **unknown**
FMTTEXT	אלמנטים עבור תבניות טקסט: **<em>**, **<strong>**, **<b>**, <**i**>, **<u>**, <**big**> ו- <**small**>
ID	שמות תואמי XML המזהים באופן ייחודי אלמנט שבמסמך
IEVENTS	אירועים פנימיים ברמת הכרטיס: **onenterforward**, **onenterbackward** ו- **ontimer**
LENGTH	שלם המציין אורך בפיקסלים, או שלם בלווית סימן קידומת, כדי לציין אורך באחוזים מרוחב המסך
NAME	שם **XML** תקף - אותיות, ספרות ותו קו-תחתון (Underscore)
NUMBER	שלם תקף הגדול או שווה לאפס
STRING	טקסט של שורה אחת ב-Unicode 2.0 שלא נותח
TEXT	טקסט של שורות מרובות ב-Unicode 2.0 שלא נותח
VDATA	**מחרוזת** עם התייחסות משתנה אפשרית
URL	**U**niform **R**esource **I**dentifier (URI), URL, **U**niform **R**esource **N**ame (URN),

URLs

WML היא שפת סימון ליצירת יישומים מבוססי Web. דרוש לה מנגנון המאפשר פנייה לתוכן ותוכניות באתרי Web. פורום WAP בחר להשתמש בסטנדרטים קיימים של אינטרנט לאספקת מנגנון זה.

WML משתמשת בכתובות URL מוחלטות כדי לפנות לתוכן המוצב באתרי Web ספציפיים, או בכתובות פרוטוקול אינטרנט (IP). בנוסף, היא משתמשת גם ב-URLs יחסיים, במקום בו ה-URL הבסיסי של מסמך WML הוא מיקום ה-Web של אותו מסמך. פנה ל-RFC 2396 להגדרות URLs וקבוצות תווי URL.

WML משתמשת גם במושג **עוגנים** (anchor), כפי שהם משמשים ב-HTML, כדי להצביע על מיקומים בתוך מסמכים. עוגן מתחיל עם URL ואחריו הסימן "#" ושם המקטע. במסמכי WML, שם מקטע הוא שמו של כרטיס מסוים בתוך המסמך. תיאור מפורט של כרטיסים ראה בסעיף Cards בפרק זה.

הקשר

בכל רגע נתון, לסוכן-משתמש תואם WML יש הקשר. ההקשר כולל את כל המשתנים המוגדרים כרגע, את ערכיהם ואת מחסנית ההיסטוריה של כל ה-URLs שהסוכן-משתמש ביקר בהם לאחרונה (עומק מחסנית ההיסטוריה תלוי ביישום הסוכן-משתמש). סוכן-משתמש WML חייב לכלול תבנית ממשק משתמש (user interface construct), דוגמת הלחצן Back בדפדפני HTML, לצורך ניווט לאחור במחסנית ההיסטוריה.

כפי שתראה, יש דרכים להגדרת משתנים ולהחלפת ערכיהם בביטויים. כמו כן, ישנן פקודות המאפשרות לך לאתר את האלמנט העליון שבמחסנית ההיסטוריה, להגיע אליו וללכת ל-URL חדש, תוך דחיפת ה-URL הנוכחי למחסנית ההיסטוריה. באפשרותך גם לנקות את המשתנים ואת המחסנית על ידי יצירת הקשר חדש.

הקשר הסוכן-משתמש הוא עקבי. הוא נשאר בהתקן במהלך תרגומי מסמכים, העברות רשת ואירועים אחרים. למרות שאין הדבר נדרש על ידי מפרט WAP, סוכן-משתמש יעיל עשוי לשמר את ההקשר במהלך כיבוי המכשיר – הדבר חוסך עלויות רשת למשתמש. כמו כן המשתנים הגלובליים ומחסנית ההיסטוריה זמינים לכל רכיבי התוכנית.

קבוצת תווים

WML משתמשת בקבוצת תווי XML, ISO/IEC-10646 שמסיבות מעשיות, זהה כרגע ל-Unicode 2.0. תוכל להשתמש בכל קבוצת-משנה מתאימה של Unicode, כגון US-ASCII או UTF-8. אם תשתמש בקבוצת-משנה אחרת מאשר US-ASCII, עליך להכריז על קבוצת התווים תוך שימוש בכותרות HTTP, ב- meta-information או בהכרזות הקדמה של XML, תוך שימוש בתכונה **encoding**. הקדמת XML מתוארת בהמשך פרק זה.

כמו XML, ‏WML תומכת בישויות תו ממוספרות לצורך פנייה לתווים מסוימים שבקבוצת תווי Unicode. ישויות ממוספרות ניתנות לביטוי בתבנית עשרונית או בתבנית הקסדצימלית. התבנית העשרונית מתחילה בקידומת "&#", התבנית ההקסדצימלית מתחילה בקידומת "&#x" (ה-x יכולה להיכתב גם כאות רישית X). הקידודים הבאים לרווח (" ") זהים:

תווים מיוחדים

ל-WML מספר תווי מיוחדים שכדי לכלול אותם בתבנית טקסט, עליך להשתמש בייצוגי תו מיוחדים כמוצג בהמשך. סימן הנקודה-פסיק (;) הוא חלק מישות התו, וחייבים להוסיף אותו. אחרת, הסוכן-משתמש מפרש את הייצוג כסדרה של תווים נבדלים, ולא כייצוג תו מיוחד.

טבלה 4.2 תווים מיוחדים

<	<	less than
>	>	greater than
'	'	apostrophe
"	"	double quote
&	&	ampersand
$	$$;	dollar sign
רווח קשיח		nonbreaking space
מקף רך		soft hyphen

רווח קשיח הוא רווח שאינו ניתן להסרה. כרגיל, סוכן-משתמש WAP מסיר רצף של תווי רווח מכל סוג שהוא.

מקף רך הוא מקף במחרוזת טקסט שאינו מוצג, אלא אם כן הוא מופיע בסופה הלוגי של שורה ששונתה על ידי הסוכן-משתמש.

טקסט

אלמנטים של WML, תכונות וערכי תכונה ממוספרים חייבים להיכתב באותיות רגילות (lower case). כל יתר תבניות השפה הן רגישות לגודל-אות (case sensitive). לדוגמה, המשתנים **Var1**, **var1** ו- **VAR1** שונים לחלוטין.

הרווח הלבן של WML כולל את התווים הבאים: שורה חדשה - new line (התו העשירי בקבוצת התווים Unicode), carriage return (התו ה- 13 ב-Unicode), רווח - space (התו ה- 32 ב-Unicode), טאב - tab (התו התשיעי ב-Unicode). מלבד היכן שצוין, WML מטפלת בתווי רווח מרובים כתו-רווח יחיד. הקוד המוסבר בספר זה משתמש

ב-CR (**c**arriage **r**eturn) וב-tab לשיפור הקריאות. הם אינם נחוצים ומהדרי WML וסוכני-משתמש מתייחסים אליהם כאל רווח לבן רגיל.

בעת הגדרת תכונות, עליך להפריד זוגות של תכונה/ערך באמצעות רווח לבן. אולם אינך יכול להשתמש ברווח לבן בין שם התכונה, הסימן שווה ("=") וערך התכונה.

הערות

הערות WML משתמשות באותה תבנית כהערות HTML:

```
<!-- הערה -->
```

ניתן לכלול אותן בכל מקום שאלמנט יכול להתרחש. לא ניתן להציב הערות באמצע הגדרת אלמנט.

גרשיים

WML מתייחסת אל תו גרש יחיד (') ואל תו גרשיים (") באופן שווה כדי לתחום ערכי תכונות. תוכל להקיף הגדרות תכונה עם אחד מסוגי התווים האלה. אם ברצונך להשתמש באחד התווים האלה במשמעותו הספרותית, עליך להקיף אותו בתווים מהסוג השני. שני האלמנטים הבאים שווים:

```
<p align="right" mode="wrap">
  פיסקה A
</p>
<p align='right'>
  פיסקה A
</p>
```

Decks

חפיסה (deck) היא יחידת ה- WML המשודרת להתקן WAP. מבחינה מושגית, החפיסה היא דף מידע, דומה מאוד לדף Web שעמו המשתמש יוצר אינטראקציה. לאורך ספר זה, אנו משתמשים לחילופין במונחים **חפיסה** ו**מסמך**.

WAP מעוצבת עבור התקנים ניידים הפועלים על רשתות סלולריות וברוחב-פס צר (בקצב של 1 kilobit לשנייה). עקב מאפייניהן של רשתות סלולריות, עדיף לנסות לשמור את מימדי החפיסות קטנים ככל האפשר, רצוי פחות מ- 1 kilobyte.

למספר שירותי WAP יש הגבלת מקסימום לגבי גודלה של חפיסה שהם מסוגלים לשדר בהעברה אחת – אילוץ הכופה עליך שמירה על גודל חפיסה קטן. אם אתה מוצא את עצמך מעצב יישום מסובך במיוחד, יהיה עליך לפרק אותו לחפיסות רבות כדי לענות על מגבלות אלו.

חפיסת WML המעוצבת כראוי מתחילה עם הקדמה, אחריה מספר כותרות אופציונליות, ואחריהן רצף של כרטיסים. כרטיס הוא יחידה אחת של אינטראקציית משתמש, כגון תפריט בחירה או מסך של טקסט. רצוי שתוכנו הוויזואלי של כרטיס יתאים במלואו למימדי מסך של התקן נייד. לעומת זאת, התוכן יכול להיות גדול ממימדי המסך, ואז ניתן לעשות שימוש בגלילה אנכית או אופקית. אם התוכן כולל מספר שדות הוספת נתונים שלא כולם מתאימים למסך, הסוכן-משתמש עשוי להציג כל אחד מהם במסך נפרד.

מכיון שיישומי WAP הם תוכניות XML, הם חייבים להתחיל עם הקדמת XML תקפה המכילה את גרסת XML ומצביע להגדרת XML של השפה בה משתמשים.

כל הדוגמאות בספר זה, הכלולות גם בתקליטור המצורף, עושות שימוש בהקדמה הבאה:

```
<?xml version="1.0"?>
<!DOCTYPE WML PUBLIC "-/WAPFORUM/DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
```

מהדרי WAP ושרתים משתמשים בהקדמה כדי לנהל את יצירתן והפעלתן של תוכניות WAP. ההתקנים המפעילים תוכניות WAP אינם נחשפים למידע זה.

לשם הקיצור, איננו כוללים את ההקדמות בקוד המוצג בספר (מלבד בדוגמה הבאה), אך הן נכללות בקבצי קוד המקור שבתקליטור המצורף.

לאחר הקדמת החפיסה, באה החפיסה הממשית, תחומה על ידי התגית <**wml**>. להלן תחביר של חפיסה. שים לב שלחפיסה אין תכונות:

```
<wml>
  <head> </head>
  <template> </template>
  <card> </card>
</wml>
```

ישנם שלושה אלמנטים התקפים בחפיסה:

- אלמנט <**head**> המכיל מידע אופציונלי אודות כלל החפיסה, כולל בקרת גישה ו- meta-information. תיארנו את אלמנט <**head**> בקצרה.
- אלמנט <**template**> המכיל מידע אופציונלי אודות אירועים קשורים ברמת החפיסה. מידע נוסף אודות אלמנט <**template**>, ראה בהמשך פרק זה.
- כרטיסים מגדירים את ממשק המשתמש ואת לוגיקת התהליך של החפיסה. חפיסה חייבת לכלול כרטיס אחד או יותר כדי להיות תוכנית WAP תקפה. אנו מתארים כרטיסים ביתר פירוט בהמשך פרק זה.

להלן דוגמת מבנה צורת חפיסה תואמת WAP 1.1 :

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
  "http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
        <head>
                מידע כותרת
        </head>
        <template>
                מפרט הגדרת תבנית
        </template>
        <card>
                מפרט הגדרת כרטיס
        </card>
        <card>
                מפרטי הגדרת כרטיס אופציונליים נוספים
        </card>
</wml>
```

לאלמנט <**head**> אין כל תכונות, אך הוא יכול להכיל שני אלמנטים נוספים. התחביר הוא כדלהלן :

```
<head>
        <access>
        <meta>
</head>
```

אלמנט <**access**> מאפשר לך להגדיר פקדי גישה, בצורה של domain ושם נתיב, עבור חפיסת WML. התחביר הוא כדלהלן :

```
<access
  domain="STRING"
  path="STRING"
>
```

domain הוא שם חלקי או שלם של domain. ערך ברירת המחדל הוא ה-domain של החפיסה הנוכחית. **path** הוא שם נתיב יחסי או מוחלט. ערך ברירת המחדל שלו הוא "/".

כאשר סוכן-משתמש מנווט בין חפיסה לחפיסה, ונתקל בחפיסה בעלת בקרי גישה, הוא משווה את domain החפיסה המבוקשת לתכונות domain ו- path של החפיסה המבוקרת.

אם ה-domain והנתיב זהים, החפיסה החדשה נטענת ומופעלת. בדיקת ההתאמה נעשית מול domain מלא ומרכיבי נתיב. לדוגמה, ה-domain של forum.org תואם ל- wap.forum.org, אך אינו תואם את www.wapforum.org. בדומה לכך, הנתיב "/X/Y" תואם את "/X" אך לא את "/XY".

בדומה לתגית <meta> של HTML, האלמנט <meta> של WML משמש לצורך שידור meta-information גנרי אודות חפיסה. להלן התחביר:

```
<meta
  http-equive="STRING"
  name="STRING"
  forua="true | false"
  content="STRING"
  scheme="STRING"
>
```

קיימים שני סוגי meta-information שבאפשרותך להגדיר, אולם ביכולתך להגדיר רק <**meta**> אחד לכל אלמנט.

הראשון, **http-equiv**, מגדיר את כותרת HTTP, במקום בו ערך **http-equiv** הוא שם הכותרת וערך ה-**content** הוא הגדרת הכותרת. WAP gateway אמור להמיר אלמנט **<meta> http-eguiv** לכותרת תגובה, לפני שליחת החפיסה אל הסוכן-משתמש. למידע נוסף אודות כותרות HTTP ו- WML, ראה פרק 8.

הסוג השני של מידע <**meta**> הוא מידע **name**, המציין שה- meta-information שצוין על ידי התכונה **content** הוא לא אחר משווה הערך לכותרת HTTP. סוכני-משתמש WAP מתעלמים מאלמנטים מסוג **<meta>name**.

התכונה **forua**, כאשר היא מוגדרת ל-**true**, מציינת שה- meta-information אמור להימסר לסוכן-משתמש, אם הסוכן-משתמש תומך באלמנט <**meta**> (אין הוא נדרש לעשות כן).

התכונה **scheme** מציינת מידע משני לצורך פירוש ה- meta-data.

לדיון מפורט יותר ב- meta-information, עליך לעיין ב-RFC 2616 לקבלת פרטים אודות כותרות HTTP, וב- HTML4 לקבלת פרטים על תגית HTML <**meta**>.

Cards

ליבה של תוכנית WML הם הכרטיסים. בדומה לתוכנית HTML, כרטיס מכיל תערובת של מידע תבנית, תוכן הניתן להצגה והוראות הפעלה. כל כרטיס בחפיסה חייב להכיל אלמנט אחד או יותר. המידע שבאלמנטים הוא משתי קטגוריות הבסיס: הוראות ותוכן.

WML מאפשרת לך לנהל משימות הרצה טיפוסיות, מרמה גבוהה למדי, כגון הוספת נתונים, ניווט ותגובה לאירועי התקן כמו הקשות מקש. תוכן הכולל טקסט בתבנית וטקסט ללא תבנית, רשימות בחירה והדרכת משתמש ועוד.

כאשר התקן תואם WAP מקבל חפיסה (או מפעיל חפיסה מבוססת **ROM** או **RAM**), הוא בוחן את האלמנטים ברמת החפיסה, מחפש את הכרטיס הראשון בחפיסה, בוחן את תכונותיו והגדרות נוספות, ואז מציג את תוכנו של הכרטיס הראשון בחפיסה

(אלא אם כן נתקבלה הוראה אחרת). הפעלת התוכנית נמשכת בתגובה לקלט משתמש או אירועים אחרים, כגון סיומו של קוצב זמן. בנקודה מסוימת, החפיסה הנוכחית עשויה להתקשר לחפיסה אחרת, ולהתחיל את המעגל מחדש.

להלן תחביר של אלמנט <**card**>:

```
<card
 id="ID"
 newcontext="true | false"
 onenterbackward="URL"
 onenterforward="URL"
 ontimer="URL"
 ordered="true | false"
 title=VDATA
>
  <onevent>, <do>, <p>, <timer>
</card>
```

לכרטיסים יש מספר תכונות, כולן אופציונליות:

- **id**. שם הכרטיס, הניתן לשימוש כמציין מקטע ב-URL.
- **newcontext**. תכונה בוליאנית המורה להתקן להסיר את כל המשתנים מצייני ההקשר, לנקות את מחסנית ההיסטוריה, ולהגדיר את ההתקן למצב ידוע.
- **onenterbackward**. URL שיש להגיע אליו אם כרטיס זה מופעל כתוצאה ממשימת <**prev**>.
- **onenterforward**. URL שיש להגיע אליו אם כרטיס זה מופעל כתוצאה ממשימת <**go**>.
- **ontimer**. URL שיש להגיע אליו אם אלמנט <**timer**> מסתיים. לפרטים נוספים, ראה סעיף "קוצבי זמן" בפרק זה.
- **ordered**. תכונה בוליאנית המציינת שתוכן הכרטיס אמור להיות מוצג בתבנית רשימה.
- **title**. התווית המשמשת לציון הכרטיס אם המשתמש סימן אותה. התווית ניתנת להצגה גם על ידי סוכני-משתמש אחרים.

במבט ראשון, נראה שאינך יכול לעשות הרבה עם כרטיס, בהתחשב במספרם המוגבל של אלמנטים שכרטיס יכול להכיל. לעומת זאת, אם נתבונן בהגדרות אלמנט WML, עליך לחשוב במונחי קינון, כרטיס יכול להכיל פסקאות (אלמנט <**p**>). פסקאות יכולות להכיל את כל שאר סוגי האלמנטים. למעשה, אינך יכול להציב כל תוכן ויזואלי בכרטיס בלי להכלילו באלמנט פיסקה. עיין בנספח א' עבור רשימה מלאה של האלמנטים הניתנים להכללה בפסקאות.

כל הדוגמאות בספר זה, הכלולות גם בתקליטור המצורף, עושות שימוש בהקדמה הבאה:

```
<?xml version="1.0"?>
<!DOCTYPE WML PUBLIC "-/WAPFORUM/DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
```

להלן מסמך WAP שלם (ללא כותרת XML, אותה חובה לציין, אך אנחנו לא נציג אותה). ה-Hello World הפופולרי תמיד, מראה כמה פשוטה חפיסת WML יכולה להיות (ex4-01.wml).

```
<wml>
  <card>
    <p>
     Hello World.
    </p>
  </card>
</wml>
```

תרשים 4.1 מציג כיצד נראית החפיסה בערכה של Nokia. כל כרטיס בחפיסה חייב להכיל אלמנט אחד או יותר הניתן לאפיון כתוכן, כהוראות טיפול באירוע, או אלמנטים של הוספת נתונים. בהמשך נדון בהרחבה בשלושת סוגי אלמנטים אלה.

תוכן

המידע הבסיסי שתוכל לכלול בכרטיס הוא מחרוזות-תוכן של תווים המוצגים על המסך של התקן תואם-WAP. כמו כן, WML מספקת מספר אפשרויות לעיצוב טקסט על המסך. כמו ב-HTML אינך יכול להגדיר את העיצוב במדויק. כל שבאפשרותך לעשות הוא לתת לסוכן-משתמש רמזים אודות מאפייני התוכן והעדפותיך. הסוכן-משתמש הוא שמחליט סופית כיצד להציג את התוכן.

```
Hello World.



OK
```

תרשים 4.1 Hello World, בסגנון WAP.

כל תוכן בכרטיס חייב להיכלל באלמנט פיסקה. להלן התחביר:

```
<p
 align="left | center | right"
 mode="wrap | nowrap"
>
 <a>, <anchor>, <br/>, DENTRY, <do>, FMTTEXT, <img/>, <table>, TEXT
</P>
```

תכונת **align** מיישרת טקסט לשמאל, לימין או למרכז התצוגה. ערך ברירת המחדל אם לא צוין ערך אחר, הוא לשמאל.

תכונת **mode** מפקחת האם תוכן הפיסקה יגלוש מצידה הימני של התצוגה. אם אתה מגדיר **nowrap**, על הסוכן-משתמש לספק מנגנון כלשהו להצגת אותו חלק של שורה בפיסקה שאינו מתאים לגבולות האופקיים של התצוגה.

אנו מציגים דוגמה של פיסקה המשתמשת בשתי תכונות אלו ב-"Aligned Text". פנה לטבלה 4.1 או לנספח א' לצורך הסברים על **FMTTEXT**, **TEXT** ו-**DENTRY**.

טקסט לא מעוצב

כרטיס יכול להכיל טקסט כלשהו או תווים מיוחדים, המהווים חלק מהשפה המופעלת בעת תצוגת הכרטיס. כאשר התקן WAP מטפל בטקסט זה. הוא מציג אותו על המסך ברצף בו הוא מופיע בכרטיס או ברצף שהוגדר באמצעות הוראות עיצוב נוספות.

בדומה לדפדפן Web, כל התקן סוכן WAP מחליף את התוכן בדרך היותר מתאימה להתקן. עבור טקסט פשוט, הכוונה היא שכל רווח לבן חיצוני מוסר, והטקסט מוגבל לגבולותיה של מילה. אם חילופי תצוגת התוכן אינם תואמים במלואם את המסך, הדפדפן מרחיב את הכרטיס אל מעבר גבולו של העמוד, והמשתמש יכול לגלול מטה את המסך תוך שימוש במקש חץ-מטה או בלחצן גלילה.

להלן דוגמה של טקסט פשוט. תרשים 4.2 מציג כיצד חפיסה זאת נראית בדפדפן Phone.com (קובץ תרגול ex4-02.wml).

```
Let's write some WAP
apps! As you can see,
white spaces gets
removed, text gets
wrapped, & special
characters are
-----------------------
 OK
```

תרשים 4.2 תצוגת טקסט פשוט

```
<wml>
  <card>
    <p>
      Let's write some WAP apps!
      As you can see,  white space gets removed, text gets wrapped,
      &amp; special characters are acknowledged.
    </p>
 </card>
</wml>
```

טקסט מיושר

WML כוללת שני אלמנטים ליישור טקסט. הראשון הוא אלמנט העצירה (**break**) בעל אפקט זהה לזה של HTML break‏ :

האלמנט **break** מתחיל שורה חדשה בתצוגה, הוא חסר כל תכונות ואינו מכיל תוכן.

ל-WML יש גם אלמנט פיסקה, כפי שכבר תיארנו. להלן דוגמה כיצד להשתמש בו :

```
<wml>
    <card>
    <p align="left" mode="wrap">
        A long left-aligned wrapping line.
    </p>
    <p align="left" mode="nowrap">
        A long left-aligned non-wrapping line.
    </p>
</card>
</wml>
```

הסוכן-משתמש הוא שמחליט כיצד לנהל טקסט ללא גלישה. דפדפן Phone.com (למשל) מאפשר למשתמש לגלול מעלה ומטה, תוך שימוש במקשי החיצים, לכל שורת טקסט המתרחבת אל מעבר לצדו הימני של המסך (לא ניתן לגלול שורות בעלות אפשרות גלישה). כאשר אתה מגיע לשורה שאינה מאפשרת גלישה, סימון יופיע בצדה השמאלי של השורה (תרשים 4.3), ולאחר הפסקה קצרה, החלק החסר הלא נראה של השורה מתגלה (תרשים 4.4).

```
A long left-aligned
wrapping line.
non-wrapping line.

OK
```

תרשים 4.4 חלקה השני של שורה ללא אפשרות גלישה

```
A long left-aligned
wrapping line.
>A long left-aligned

OK
```

תרשים 4.3 חלקה הראשון של שורה ללא אפשרות גלישה

טקסט מעוצב

בדומה ל-HTML, WML כוללת מיגוון אלמנטים לעיצוב טקסט. כמו ב-HTML הפירוש שלהם מושאר לדפדפן. להלן טבלה של תגיות סגנון של טקסט WML:

טבלה 4.3 תגיות סגנון

תגית	מאפייני גופן
<b>	Bold
<big>	Large
<em>	Emphasize
<i>	Italic
<small>	Small
<strong>	Strong Emphasis
<u>	Underline

בשל טבעם הכללי יותר, מעודד תקן WML את המפתחים להשתמש באלמנטים <**strong**> ו-<**em**> היכן שניתן. סוכני-משתמש WAP יכולים להתעלם מתגיות סגנון, ולהחליף את הטקסט ללא סגנון.

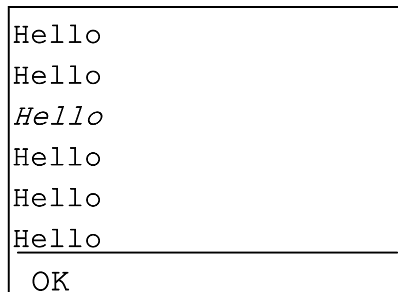
כל אחת מתגיות סגנון הטקסט היא בעלת תחביר בסיסי זהה. אין להן תכונות, והן יכולות להכיל את הישויות הבאות:

<a>, <anchor>,
, FMTTEXT, <img/>, <table>, TEXT

להלן חפיסה פשוטה המשתמשת באלמנטים שונים של סגנון טקסט (ex4-04.wml) :

```
<wml>
  <card>
    <p>
      <big>Hello</big><br/>
      <em>Hello</em><br/>
      <i>Hello</i><br/>
      <small>Hello</small><br/>
      <strong>Hello</strong><br/>
      <u>Hello</u><br/>
   </p>
  </card>
</wml>
```

תרשים 4.5 מציג כיצד חפיסה זו נראית (לא כל הדפדפנים תומכים בגופן מוטה).



תרשים 4.5 אלמנטי התבנית של WML

טבלאות

בנוסף לתכונות יישור פיסקה רגילות שתוארו לעיל, ל-WML יש שלושה אלמנטים נוספים - table, table row ו- table data - המאפשרים לך להגדיר טבלה מורכבת יותר. טבלאות WML דומות לטבלאות HTML, אך עם פחות אפשרויות עיצוב. לדוגמה, ביכולתך לשלוט ביישור היחסי של העמודות, אך לא במיקום המדויק של כל עמודה בתצוגה.

להלן תחביר טבלה :

```
<table
  align="STRING"
  columns="NUMBER"
  title="VDATA"
>
 <tr></tr>
</table>
```

מחרוזת **align** היא רשימת מצייני יישור – אחד לכל עמודה. ערכים תקפים הם "L" (שמאל), "C" (מרכז) ו- "R" (ימין). התכונה הנדרשת **columns**, מציינת את מספר העמודות שבטבלה. **title** הוא שמה של הטבלה. הסוכן-משתמש יכול להשתמש ב- **title** כאשר הוא מציג את הטבלה.

כדוגמת טבלאות HTML, הגדרת תכולת הטבלה נעשית על ידי קינון של הצהרת נתונים אחת או יותר בתוך שורת הצהרות. נתוני טבלה יכולים להיות טקסט פשוט, או טקסט תבניתי, תמונות או עוגנים. הסוכן-משתמש משבץ את הנתונים בתאי הטבלה, ומציג את התוצאה.

שורת טבלה יכולה להכיל אלמנט טבלה אחד או יותר, המספר מצוין בתכונת **columns** של הטבלה. להלן תחביר של שורת טבלה:

```
<tr>
   <td></td>
</tr>
```

אלמנט נתוני הטבלה מורכב מעט יותר:

```
<td>
  <a>, <anchor>, <br/>, FMTTEXT, <img/>, <table>, TEXT
</td>
```

אלמנט נתוני טבלה יכול להכיל את רוב סוגי התוכן, כולל תמונות, עוגנים וטבלאות אחרות.

להלן דוגמת טבלה פשוטה (ex4-05.wml):

```
<wml>
  <card>
    <p align="center">
      <i> Appointments </i>
    </p>
  <p>
     <small>
       <table columns="2">
           <tr> <td> Date </td> <td>   Start/Stop </td> </tr>
           <tr> <td> 02/25 </td> <td> 9:30-10:45A </td> </tr>
           <tr> <td> 03/03 </td> <td> 4:45-6:00P </td> </tr>
     </table>
    </small>
  </p>
 </card>
</wml>
```

תרשים 4.6 מציג כיצד תיראה טבלה זו.

```
      Appointments
Date      Start/Stop
02/25     9:30-10:45A
03/03     4:45-6:00P

OK
```

תרשים 4.6 טבלת WAP פשוטה

תמונות

בעידן הנוכחי של אתרי Web עתירי גרפיקה, ייתכן שייראה מוזר להציג תמונות בהתקן מוגבל יכולת, כגון התקן תואם WAP. ממשקי משתמש גרפיים עשירים אפשריים על התקנים ניידים. תמונה שווה מילים לא מועטות, ויכולה לתפוס שטח תצוגה קטן הרבה יותר.

מעצבי WAP כללו אלמנט תמונה, המשתמש בתגית **<img/>**, עבור אותם מצבים שעבורם תמונה היא הבחירה הטובה יותר. תמונות מוצגות בתוך זרם הטקסט הרגיל. ישנן תכונות שהסוכן-משתמש יכול לעשות בהן שימוש, אם הוא תומך בהן. הדבר מקנה לך שליטה מסוימת על אופן תצוגת התמונה. לעומת זאת, אם ההתקן מתעלם מרמזים אלה, כל אשר ביכולתך לעשות הוא להציב את התמונה בשורה נפרדת משלה.

להלן התחביר עבור התגית **<img/>**:

```
<img
  align="top | middle | bottom"
  alt="VDATA"
  height="LENGTH"
  hspace="LENGTH"
  localsrc="VDATA"
  src="URL"
  vspace="LENGTH"
  width="LENGTH"
/>
```

לאלמנט התמונה מספר תכונות, אולם רק שתיים דרושות: **alt** ו-**src**. התכונה **alt** היא מחרוזת הניתנת להצבה במקום התמונה. הדבר נחוץ עבור התקנים שאינם תומכים בגרפיקה, או שהתמונה אינה ניתנת להשגה. התכונה **src** היא ה-URL הממשי של התמונה. בדיוק כמו דפדפן HTML, סוכן-משתמש WAP, בדרך כלל משיג קודם כל את החפיסה המכילה את התגית **<img/>**, וכפעולה נפרדת דורש את התמונה המוצבת ב-URL הרשום בתכונה **src**.

התכונה **localsrc** מציינת שם תמונה מבוססת ROM (או RAM), הניתנת לשימוש במקום התמונה המוצבת ב-URL **src**. מפרט WML 1.1 קובע שתמונות **localsrc** הנן בעלות עדיפות על תמונות **src**, ויש להשתמש בהן אם הן קיימות. הדבר הגיוני, מכיון שתמונות מקומיות נטענות מהר יותר, ומונעות פעולת השגת נתונים נפרדת. סוכן-משתמש כולל, בדרך כלל, רשימה נרחבת של סמלים ותמונות מבוססי-ROM למטרה זו.

להלן חפיסה פשוטה המציגה את דוח מזג האוויר (ex4-06.wml):

```
<wml>
  <card>
     <p>
         Today's forecast:
         <br/>
          <img
            alt="Partly cloudy"
            src="./pics/pcloudy.wbmp"
            localsrc="partcloudy"
         />
       <br/>
     </p>
   </card>
 </wml>
```

אם הסוכן-משתמש אינו תומך בגרפיקה, תראה את התצוגה שבתרשים 4.7. אם הסוכן-משתמש חסר כל תמונות פנימיות, תראה את תמונת השחור-לבן המאוחסנת ב-URL שכתובתו "/pics/pcloudy.wbmp" במקום המחרוזת "Partly cloudy". ראה את התצוגה שבתרשים 4.8.



הערה!

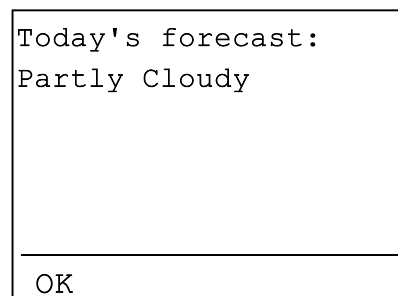
פורום WAP הגדיר תבנית תמונה משלו, ה-Wireless Bitmap Format (WBMP), האופטימלית לשידור יעיל ברשתות רוחב-פס נמוך. כל ה-WAP Compatible gateways חייבים להכיר בתבנית WBMP כסוג MIME לגיטימי. כדי לפשט פיתוח תוכן, רוב WAP gateways ממירים אוטומטית קבצי BMP לקבצי WBMP.

התכונות הנותרות בתגית <**img/**> הן אופציונליות. **vspace** ו-**hspace** הם רווחים לבנים האמורים להתווסף מעל לתמונה או מתחתיה (**vspace**), ולימינה או לשמאלה (**hspace**). ניתן לבטא את שני הערכים כ-integer, כציון פיקסלים, או על ידי integer ובעקבותיו סימן האחוז (%) המציין את אחוז הרווח הלבן במימד המתאים.

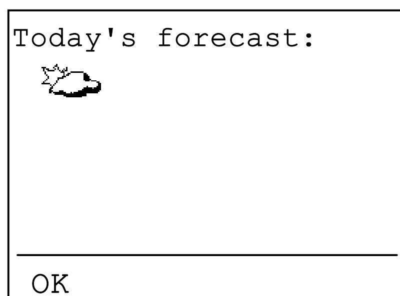
התכונה **align** מציעה יישור תמונה יחסית לשורת הטקסט הנוכחית: מעלה, למרכז, או עם שורת הבסיס הנוכחית.

התכונות **height** ו-**width** מקנות לסוכן-משתמש רמזים אודות מימדי התמונה, כך שיוכל לשמור שטח עבורה, ולהמשיך ולהחליף את התצוגה הנוכחית בעודו מוריד את התמונה. כדוגמת התכונות **hspace** ו-**vspace**, התכונות **height** ו-**width** יכולות להיות מצוינות כפיקסלים מוחלטים, או כאחוזים ממימדי התצוגה. סוכני-משתמש יכולים לקבוע את מימדי התמונה לערכי הגובה והרוחב, אם רצונם בכך.

התכונות האופציונליות הן רק רמזים לסוכן-משתמש וניתן להתעלם מהן. כמו כן, הן פתוחות לפרשנויות. אין לצפות שבהכרח שני סוכני-משתמש בעלי אותם מימדי מסך יטפלו באותן תמונות בדרך זהה.



תרשים 4.7 סוכן-משתמש ללא כל תמיכה גרפית



תרשים 4.8 דוח מזג אוויר גרפי

משתנים

שלא כ-HTML, ניתן להגדיר משתנים בחפיסות WML, להקצות להם ערכים, להציג אותם על המסך ולהשתמש בהם בביטויים. התועלת הרבה שמשתנים מקנים לך, היא היכולת לשמר מידע מצב בין חפיסות, והקלה על העברת מידע מחפיסה אחת לאחרת.

משתני WML הם חסרי-סוג. הם כולם מחרוזות שערכן מוגדר לרצף של תווים או לערך null. משתנים שלא הוקצה להם הערך, גם הם מקבלים את הערך null. ניתן להחליף ערכי משתנים בכל מקום שבו מותר טקסט. החלפה זו היא בעלת עדיפות הניתוח הגבוהה יותר בכל פעולות WML בזמן ריצה. למרות זאת, פעולה זו מבוצעת מאוחר ככל הניתן, כדי לוודא שפעולות המגדירות משתנים כבר בוצעו.

שמות משתנים הם רגישי גודל-אות, עליהם להתחיל בתו קו תחתון ("_"), או באות US-ASCII ובעקבותיה אות US-ASCII אחת או יותר, מספרים או קו תחתון. להלן שמות משתנה תקפים:

URL_name   _Var1   A100200

הדרך הפשוטה להגדיר ערכי משתנה היא להשתמש באלמנט **<setvar>**:

```
<setvar
  name="VDATA"
  value="VDATA"
/>
```

ל-<**setvar**> יש שתי תכונות: **name** (שם המשתנה) ו- **value** (הערך שברצונך להקצות למשתנה). <**setvar**> משלבת הצהרת משתנה והקצאת ערכים בצעד אחד. לדוגמה:

```
<setvar name="var1" value="some value" />
```

הכרזת המשתנה **var1** והקצאת הערך "some value" אליו. תוכל להשתמש ב-<**setvar**> באלמנטים <**go**>, <**prev**> ו-<**refresh**>, כדי לאתחל משתנים לפני ביצוע משימה (לפרטים נוספים, ראה סעיף "משימות" בהמשך).

ביכולתך להשתמש גם באלמנטים <**input**>, <**select**> ו-<**postfield**> כדי להכריז על משתנים ולהקצות להם ערכים, או להקצות להם ערכי ברירת מחדל. האלמנט <**input**> משמש להוספת נתונים. האלמנט <**select**> מאפשר למשתמשים לבחור פריט אחד או יותר מקבוצת בחירה. האלמנט <**postfield**> מתואר ביתר פירוט בהמשך פרק זה.

ניתן להתייחס למשתנה הנמצא בתוכן של כל אלמנט WML, וכמו כן בערך של תכונות מסוימות. כאשר תעשה זאת, ערך המשתנה מתחלף לטקסט החפיסה.

קיימות שלוש דרכים נפרדות שדרכן ביכולתך להתייחס למשתנים. כולן מתחילות עם תו הדולר ("$"):

```
$varname
$(varname)
$(varname:conversion)
```

הצורה הראשונה קבילה, אם אין דו-משמעות בהקשרו של שם המשתנה. אם קיימת דו-משמעות, עליך השתמש בצורה השנייה או הצורה השלישית.

מכיון שכל התייחסויות המשנה חייבות להתחיל עם "$", עליך להשתמש בשני סימני דולר כדי לייצג סימן דולר יחיד במחרוזת או בטקסט. בכרטיס הבא, אם המשתנה **amt** מוגדר ל-"10.35", המחרוזת "Your current balance is $10.35" נראית על המסך:

```
<card>
   <p>
        your current balance is $$$amt.
   </p>
</card>
```

כרטיס זה זהה לקודם:

```
<card>
   <p>
        your current balance is $&#x24;$(amt).
  </p>
</card>
```

ניתן להמיר משתנים, כאשר הם מוחלפים בתוכניות WML, המשתמשות ב-URL-escaping rules המוגדרים ב- RFC 2396. חוקים אלה מספקים מנגנון להטבעת תווים מיוחדים, כגון תו הנקודותיים (":") בלי שיפורשו כחלק מה-URL. למרות שחוקים אלה פורשו עבור התייחסות URL, תוכל ליישם אותם בכל התייחסות למשתנה.

תווי חילוף מקודמים בסימן האחוז ("%"), ובעקבותיו שני תווים הקסדצימליים המגדירים את הקוד האוקטלי של תווי החילוף של השפה הנוכחית. לדוגמה, המחרוזת:

"Out #^?!~* spot !"

מומרת ל:

"Out+%23^%3F!~*+spot !"

בעת חילוף, הצורה הראשונה אינה מתאימה להיכלל ב-URL כחלק מהגדרת משתנים, ואילו הצורה השנייה מתאימה. טבלה 4.4 מציגה את תווי החילוף מ- RFC 2396.

כאשר ערך מחרוזת-משתנה מוחלף בחפיסת WML ביכולתך לציין תווי חילוף, תווים שאינם ניתנים לחילוף, או לא מוחלפים, תוך שימוש בתחביר כדלהלן:

טבלה 4.4 URL-escape characters

DELIMITERS	UNWIZE	RESERVED
%3c <	%7b {	%3b ;
%3e >	%7d }	%2f /
%23 #	%7c l	%3f ?
%25	55c% \	%3a :
%22	%5e ^	%40 @
	%5b [	%26 &
	%5d]	%3d =
	%27 '	%2b +
		%24 $
		%2c ,
		%20 space

תווי US-ASCII, 0x00-0x1F ו- Ox7F גם הם שמורים. הם מומרים ל- %00-%1f ו- 7f בהתאמה.

$(var:e) $(var:E) $(var:escape)   <!-- escaping -- >
$(var:u) $(var:U) $(var:unesc)   <!-- unescaping -- >
$(var:n) $(var:N) $(var:noesc)   <!-- no escaping -- >.

WML מיישמת החלפה באופן אוטומטי בעת פעולה עם תכונות המוגדרות ל-URLs. כתוצאה מכך, כמעט בכל הנסיבות עליך להתעלם ממשתני החלפה, ולהניח שהסוכן-משתמש WML עושה את הדבר הנכון.

משימות

עד כה, דנו רק בתוכן היכול להיכלל במסמכי WAP. כפי שכל מתכנת יודע, תוכנית אינה באמת תוכנית, אלא אם כן היא כוללת הוראות. קיימות מספר דרכים להגדרת תהליך בחפיסת WML. המקום הנכון יותר להתחיל בו הוא הסברתן של משימות.

WAP 1.1 מגדירה מספר סוגי משימות. משימות משפיעות על סדר הפעלתן של תוכניות WAP על ידי הגדרת פעולות שיש לנקוט בהן כתגובה לאירועים. ישנם ארבעה סוגים של משימות WML: <**noop**>, <**prev**>, <**refresh**> ו-<**go**>.

המשימה הפשוטה יותר היא <**noop**>. להלן התחביר שלה:

```
<noop/>
```

כפי שציפית, אין היא עושה דבר, היא משמשת רק לדריסה (או ביטול) של הגדרות אירוע ברמת החפיסה.

<**prev**> משמשת לצורך ניווט לכרטיס קודם שבמחסנית ההיסטוריה של הסוכן-משתמש:

```
<prev>
   <setvar/>
</prev>
```

משימת <**refresh**> יוזמת רענון התוכן הוויזואלי הנוכחי של הסוכן-משתמש:

```
<refresh>
   <setvar/>
</refresh>
```

אלמנטים של <setvar> הם בעלי קדימות הפעלה. אם הכרטיס הנוכחי מכיל קוצב זמן, אז קודם מופעלת התגית <setvar> והאלמנטים שלה ולאחר מכו מופעל הכרטיס.

האלמנט <**go**> מגדיר ניווט ל-URL. URL זה יכול להצביע על חפיסת WML חדשה המאוחסנת בשרת, או להצביע לכרטיס אחר בחפיסה הנוכחית. האלמנט <**go**> מקנה ל-WML הרבה מכוחה וגמישותה. להלן התחביר של אלמנט <**go**>:

```
<go
 accept-charset="STRING"
 href="URL"
 method="post | get"
 sendreferer="true | false"
>
 <postfield>, <setvar>
</go>
```

href היא תכונה נדרשת. היא ה-URL של הכרטיס הבא או החפיסה אותה יש להשיג ולהציג. אם זה URL מוחלט או יחסי תקף, הניתן להשגה, ה-URL נדחף אל מחסנית ההיסטוריה, והכרטיס הראשון של החפיסה החדשה מוצג. אם ה-URL הוא עוגן (anchor) חלקי לכרטיס אחר בחפיסה הנוכחית, מחסנית ההיסטוריה נשארת ללא שינוי, והכרטיס החדש מוצג אם הוא קיים.

אם ערך **sendreferer** הוא **true**, הסוכן-משתמש חייב לציין ל- WAP gateway, תוך שימוש בכותרת דרישת "Referer" של HTTP, את ה-URL של החפיסה הנוכחית, תוך שימוש ב-URL היחסי הקטן ביותר. תוכנה זו עוצבה כדי להקנות לשרתים רמה מסוימת של בקרת גישה ל-URLs, בהתבסס על זהות החפיסות המתייחסות אליהם. הגדרת ברירת המחדל שלה הוא **false**.

method מתייחסת לשיטת ייחוס HTTP המשמשת את URL זה. שתי הבחירות האפשריות הן **get** ו-**post**. הן מפעילות בהתאמה את דרישות GET ו- POST של שרת HTTP. הדרישות כוללות את כל הכרזות השדה הנכללות בגוף האלמנט <**go**>. הגדרת ברירת המחדל של **method** היא **get**.

accept-charset היא רשימת exclusive-OR של קבוצת תווים תקפה, כולל שמות שהוגדרו ב- RFC 2045 ו- RFC 2616. השרת המקבל את הדרישה שיוצרה על ידי משימת <**go**> זו חייב לקבל את אחד מפרטי הרשימה. ערך ברירת המחדל של **accept-charset** הוא **unknown**, המציין לסוכן-משתמש שעליו להשתמש באותה קבוצת תווים ששימשה לשידור החפיסה אל ההתקן.

להלן הגדרת משימת <**go**> פשוטה:

```
<go href="./new-deck.wml" sendreferer="true" />
```

אלמנטי <**postfield**> משמשים לצורך הגדרת זוגות של name/value (שם/ערך) המועברים לשרת HTTP, המקבל את דרישת <**go**>. להלן תחביר **postfield**:

```
<postfield
  name="VDATA"
  value="VDATA"
/>
```

כאשר המשימה בעלת אלמנטי <**postfield**> מופעלת, הסוכן-משתמש:

1. מזהה את זוגות ה-name/value, ומחליף משתנים.
2. מעביר את זוגות ה-name/value לקבוצת התווים הנכונה.
3. מחליף את זוגות ה-name/value בהתאם לחוקי ההחלפה של ה-URL, ומרכיב אותן ליישום בצורת קוד x-www URL עבור סוג תוכן MIME.
4. משלים את המשימה בהתבסס על התכונה **method**.

הגדרות תכונת **method** קובעות כיצד הערכים נדחפים אל השרת.

אם הגדרת את **method** כ-**get**, זוגות ה-name/value מוספים לחלק השאילתה של דרישת HTTP, והדרישה נשלחת ל-URL המצוין. לדוגמה, ניתן להשתמש במשימת <**go**>. זו לצורך דרישת רשימת כל הטיסות שבין San Francisco ל-Chicago ביום שני הבא :

```
<go href="/flights.cgi" sendreferer="true"  method="get">
  <postfield name="day" value="Mon" />
  <postfield name="origin" value="SFO" />
  <postfield name="destination" value="ORD" />
</go>
```

הדבר מבטיח את דרישת **GET** HTTP הבאה :

```
GET /flights.cgi?day=Mon&origin=SFO&destination=ORD HTTP/1.1
'
' other HTTP headers
'
```

אם אתה מגדיר את method כ-**post**, אותה משימת <**go**> שולחת את דרישת HTTP **POST** זו :

```
POST /flights.cgi HTTP/1.1
content-type='xxx-urlencoded'
'
' other HTTP headers
'
day=Mon&origin=SFO&destination=ORD"
```

להסברים מפורטים נוספים אודות דרישות **GET** ו-**POST** וכותרות **HTTP**, ראה פרק 8.

אירועים

משימות אינן פועלות בריק. כדי לבצע דבר מה מועיל, הן חייבות להיות קשורות לאירוע. כאשר האירוע מתרחש, המשימה מופעלת. ישנם שלושה אלמנטים שונים המשמשים לקשירת משימה לאירוע : <**anchor**>, <**onevent**> ו- <**do**>.

עוגנים (Anchors)

כמו ב-HTML, תוכניות WML יכולות לכלול עוגנים – קישורים לאלמנטים שבמסגרת הדף. העוגן מכיל תוכן, הנראה בתצוגת ההתקן באופן כזה שהמשתמש יודע שאכן קיים שם עוגן.

ב-HTML, עוגנים בדרך כלל מודגשים בקו תחתון ובצבע שונה מהתוכן הרגיל. לסוכני-משתמש תואמי WML אין כללים נוקשים להצגת עוגנים. על הסוכן רק להבדיל את העוגן מהתוכן הלא מעוגן. לדוגמה, דפדפן Phone.com מקיף טקסט-עוגן בסוגריים מרובעים.

להלן תחבירו של אלמנט <**anchor**>:

```
<anchor
  title="VDATA"
>
<br/>, <go>, <img/>, <prev>, <refresh>, TEXT
</anchor>
```

לעוגנים יש תכונת **title** (כותרת) אופציונלית. האופן בו מוצגת תכונת **title** נתונה לשיקוליו של הסוכן-משתמש. מפרט WML 1.1 מזכיר כאפשרויות ביצוע: תיאורי כלי (tool tips), תוויות לחצן דינמיות והוראות קוליות. ביכולתו של סוכן-משתמש להתעלם מכותרת ה-<**anchor**>. אם כותרות מוצגות, הן מוצגות בנוסף על תוכנו הטקסטואלי של העוגן – הטקסט **חייב** להיות מוצג, הכותרת **יכולה** להיות מוצגת. מפרט WML 1.1 מציע שאורכה של תכונת **title** של <**anchor**> יהיה בן שישה תווים או פחות, כך שתוכל לפעול על טווח רחב של התקנים.

בנוסף לתוכן ויזואלי כלשהו, <**anchor**> תקף חייב לכלול הגדרת משימה, המציינת לסוכן-משתמש מה עליו לעשות אם העוגן נבחר על ידי המשתמש (כיצד המשתמש בוחר עוגן, תלוי במנגנון של הסוכן-משתמש). המשימה המוגדרת חייבת להיות אלמנט <**go**>, <**prev**> או <**refresh**>.

ביכולתך להטביע עוגנים בכל מקום בתוכנית WML בו ניתן לכתוב טקסט מלבד באלמנטים מסוג <**option**> (ראה סעיף "בחירות" בהמשך, לדיון על אלמנט <**option**>).

כמו כן קיים תחביר קצר עבור עוגנים:

```
<a
  href="URL"
  title="label"
>
 <br/>, <img/>, TEXT
</a>
```

התחביר משתמש בתגית <**a**> במקום בתגית <**anchor**>, וניתן להשתמש בה רק לצורך הגדרת משימות <**go**> הדורשות ציון URL.

בעזרת ההוראות הבסיסיות למשימות, אירועים ועוגנים, תוכל ליצור כעת חפיסות WML בעלות יכולות ניווט (ex4-07.wml):

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
 "http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>

<!-- the first card -->

 <card id="card1" >
```

```
    <onevent type="onenterforward">
      <refresh>
        <setvar name="origin" value="Card # one" />
      </refresh>
    </onevent>
    <onevent type="onenterbackward">
      <refresh>
        <setvar name="origin" value="Card # one" />
      </refresh>
    </onevent>

    <p>
      Select a destination:<br/>
      <a title="Card 2" href="#card2">
        Card # two
      </a>
      <br/>
      <a title="Display" href="#card3">
        Card # three
      </a>
    </p>
  </card>

<!-- the second card -->

  <card id="card2" >
    <onevent type="onenterforward">
      <refresh>
        <setvar name="origin" value="Card # two" />
      </refresh>
    </onevent>
     <onevent type="onenterbackward">
      <refresh>
        <setvar name="origin" value="Card # two" />
      </refresh>
    </onevent>
   <p>
      Select a destination: <br/><br/>
      <a title="Card 1" href="#card1">
        <img src="./pics/uparrow.bmp" alt="Card # one " localsrc="uparrow1"/>
      </a>
      <br/>
      <a title="Display" href="#card3">
        <img src="./pics/downarrow.bmp" alt="Card # three" localsrc="downarrow1" />
```

```
    </a>
   </p>
  </card>

<!-- the third card -->

  <card id="card3" >
   <p>
    You've just come from $origin <br/>
    Select a destination: <br/>
    <a title="Card 1" href="#card1">
     Card # one
    </a>
    <br/>
    <a title="Card 2" href="#card2">
    </a>
   </p>
  </card>
</wml>
```

לחפיסה זו שלושה כרטיסים. כאשר היא מופעלת, מתגלה הכרטיס הראשון כמתואר בתרשים 4.9. אם אתה עושה שימוש בסוכן-משתמש של Phone.com שים לב כיצד הוא מדגיש עוגנים בעזרת סוגריים מרובעים. אם אתה בוחר את העוגן המביא אותך לכרטיס השני (**card2**), ישנן שתי תוצאות אפשריות. אם הסוכן-משתמש איתר סמלים ששמותיהם **uparrow1** ו-**downarrow1**, תראה תצוגה הדומה לזו שבתרשים 4.10. אם הסמלים אינם קיימים, תראה את התצוגה שבתרשים 4.11. לבסוף, אם תנוע לכרטיס השלישי, תראה את התצוגה שבתרשים 4.12, כולל שם הכרטיס ממנו הגעת.

```
Select a destination:
> [ ⇧ ]
  [ ⇩ ]

Card 1
```

תרשים 4.9 הכרטיס הראשון

```
Select a destination:
> [ Card # two ]
  [ Card # three ]

Card 0
```

תרשים 4.10 הכרטיס השני עם סמלים

```
You've just come from
Card # two
Select a destination:
> [ Card # one ]
  [ Card # two ]
_____________________
Card 1
```

תרשים 4.11 הכרטיס השני עם עוגני טקסט

```
Select a destination:
> [ Card # one ]
  [ Card # three ]
_____________________
 OK
```

תרשים 4.12 הכרטיס השלישי

אירועים פנימיים

WML מגדירה קבוצה של אירועים פנימיים – אירועים המופעלים על ידי תהליכים פנימיים של הסוכן-משתמש. ל-WML יש ארבעה סוגים של אירועים פנימיים:

- **oneventforward**. מופעל כאשר המשתמש מנווט אל כרטיס דרך משימת <**go**> או כל מנגנון אחר, כגון פונקציית תסריט, בעלת אפקט הזהה.
- **oneventbackward**. מופעל כאשר המשתמש מנווט אל כרטיס דרך משימת <**prev**> או כל מנגנון אחר, כגון הפעלת מנגנון ה-<**prev**> של ההתקן, באמצעות לחיצה על מקש **back**, בעל האפקט הזהה.
- **ontimer**. מופעל כאשר<**timer**> (קוצב-זמן) מוגדר-תוכנית מסתיים. ניתן להגדיר קוצבי זמן על ידי שימוש באלמנט <**timer**>.
- **onpick**. מופעל כאשר המשתמש בוחר או מבטל בחירה של פריט <**option**>.

קישור אירוע פנימי למשימה נעשה באמצעות האלמנט <**onevent**>. התחביר של האלמנט <**onevent**> הוא פשוט:

```
<onevent
   type="oneventforward | oneventbackward | ontimer | onpick"
>
   <go>, <noop>, <prev>, <refresh>
</onevent>
```

להלן דוגמה פשוטה העושה שימוש באירוע פנימי (ex4-08.wml):

```
<wml>
  <card>
    <onevent type="oneventforward" >
      <refresh>
        <setvar name="var1" value=" " />
        <setvar name="var2" value=" " />
      </refresh>
    </onevent>
```

```
    <p>
      Now we're starting with a clean slate!
    </p>
  </card>
</wml>
```

כל פעם שנכנסים לכרטיס, המשתנים **var1** ו-**var2** מאותחלים בערכי null. ערכים אלה אינם מוגדרים מחדש אם מנווטים אל הכרטיס שלא באמצעות משימת <**prev**> – לחצן **back** שבהתקן תואם WML, או כל פעולה אחרת שמאתרת את מיקום הכרטיס על ידי חילוצו ממחסנית ההיסטוריה.

קוצבי זמן (Timers)

קוצבי זמן מספקים מנגנון המאפשר הפעלת משימה לאחר פרק זמן מסוים. כל פעולה שמפעילה כרטיס מפעילה גם את קוצב הזמן שלו. כאשר זמן ה-<**timer**> (קוצב הזמן) פג, המשימה שקשורה אליו מתחילה. אם זרם הפעילות עוזב את הכרטיס לפני תום זמנו של ה-<**timer**>, פעולתו נפסקת. למעשה, טווח ה-<**timer**> מוגבל לכרטיס שבו הוא מוגדר. לכרטיס יכול להיות רק <**timer**> אחד, ול-<**timer**> אחד יכולה להיות רק משימה אחת.

להלן תחבירו של אלמנט <**timer**> :

```
<timer
  name="NAME"
  value="VDATA"
/>
```

התכונה **name** האופציונלית מציינת את שם המשתנה המכיל את ערך תחילת הספירה לאחור שבה מופעל ה-<**timer**>, ואת שארית הזמן כאשר מופסקת או מסתיימת פעולתו. אם **name** קיים והוא מאותחל לערך מספרי חיובי, הוא משמש כערך התחילי של קוצב הזמן, ודורס את התכונה **value**.

התכונה **value**, מכילה את ערך ברירת המחדל התחילי, בעשיריות שנייה. בערך זה נעשה שימוש אם לא קיימת התכונה **name**, או אם המשתנה המוגדר בתכונה **name** אינו מאותחל. מתעלמים מהתכונה **value** אם התכונה **name** קיימת ומאותחלת.

התוכנית הבאה מציגה סדרת תמונות (ראה תרשימים 4.13 עד 4.15). קוצבי זמן משמשים לצורך עדכון התמונה כל שתי שניות (ex4-09.wml) :

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
  "http://www.wapforum.org/DTD/wml_1.1.xml">

<wml>
```

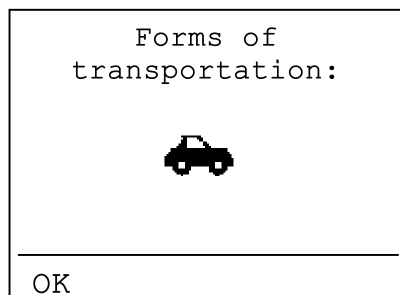
```
<!-- the automobile card -->

  <card id="card1">
    <onevent type="ontimer" >
      <go href="#card2"/>
    </onevent>
    <timer value="20"/>
    <p align="center">
      Forms of transportation:<br/><br/>
      <img src="./pics/car.wbmp" alt="Automobiles" localsrc="car"/>
    </p>
  </card>

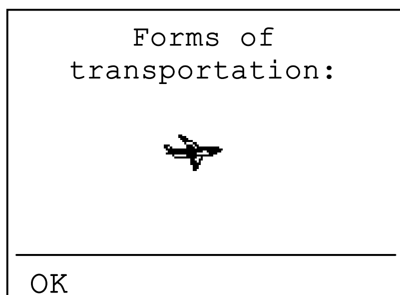
<!-- the airplane card -->

  <card id="card2">
    <onevent type="ontimer" >
      <go href="#card3"/>
    </onevent>
    <timer value="20"/>
    <p align="center">
      Forms of transportation: <br/><br/>
      <img src="./pics/airplane.wbmp" alt="Airplanes" localsrc="plane"/>
    </p>
 </card>

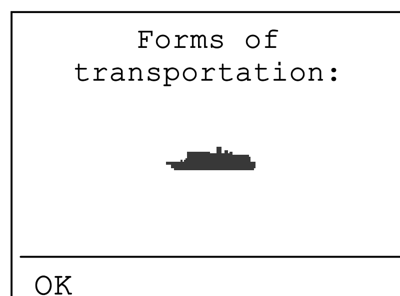
<!-- the ship card -->
  <card id="card3">
    <onevent type="ontimer">
      <go href="#card1"/>
    </onevent>
    <timer value="20"/>
    <p align="center">
      Forms of transportation: <br/><br/>
      <img src="./pics/ship.wbmp" alt="Ships" localsrc="boat"/>
    </p>
  </card>
</wml>
```



תרשים 4.13 כרטיס המכונית



תרשים 4.14 כרטיס המטוס



תרשים 4.15 כרטיס האנייה

אתה ודאי זוכר, מהחלק שעסק בתכונות כרטיס, את תכונת הכרטיס האופציונלית **ontimer**. היא שוות ערך לאלמנט <**go**>.

להלן דוגמת התוכנית הקודמת, המשתמשת בתחביר הצורה הקצרה של **ontimer** (ex4-10.wml).

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
  "http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<!-- the automobile card -->

  <card id="card1" ontimer="#card2">
    <timer value="20"/>
    <p align="center">
      Forms of transportation:
      <img src="./pics/car.wbmp" alt="Automobiles" localsrc="car"/>
    </p>
  </card>
```

```
<!-- the airplane card -->

  <card id="card2" ontimer="#card3">
    <timer value="20"/>
    <p align="center">
      Forms of transportation:
      <img src="./pics/airplane.wbmp" alt="Airplanes" localsrc="plane"/>
    </p>
 </card>

<!-- the ship card -->

  <card id="card3" ontimer="#card1">
    <timer value="20"/>
    <p align="center">
      Forms of transportation:
      <img src="./pics/ship.wbmp" alt="Ships" localsrc="boat"/>
    </p>
 </card>
</wml>
```

אירועים מופעלי-משתמש

כל התקן תואם WAP הוא בעל קבוצת כלי ממשק משתמש (user interface widget) מוגדרים מראש, הזמינים למשתמש. כלי ממשק אלה עשויים להיות לחצנים אמיתיים של טלפון, סמלים על מסך רגיש-מגע, פקודה מופעלת-קול, או כל רכיב ממשק מוכר אחר לפי לשיקול דעתם של המעצבים.

מפרט ה- WML 1.1 מגדיר את כלי הממשק הבסיסיים הבאים שכל ההתקנים תואמי-WAP חייבים לתמוך בהם. שים לב שרק אחד מכלי ממשק אלה – **prev** – הוא בעל פעולה מוגדרת מראש. כל היתר מוגדרים לפי הקשרם. מפתחי היישום הם שמחליטים מה למעשה יקרה, כאשר אחד מכלי ממשק אלה יופעל.

- **accept**. הכרה חיובית.
- **prev**. ניווט חזרה במחסנית ההיסטוריה.
- **help**. בקשת עזרה (אפשרית) רגישת-הקשר.
- **reset**. הגדרה מחדש של הקשר ההתקן.
- **options**. דרישה רגישת-הקשר עבור אפשרויות או פעולות נוספות.
- **delete**. מחיקת הפריט או הבחירה הנוכחיים.
- **unknown**. אלמנט <**do**> גנרי.

כאשר משתמש מפעיל אחד כלי ממשק אלה, הוא יוצר אירוע שביכולתך לאתר ולהגיב עליו, תוך שימוש באלמנט WML <**do**>. להלן תחביר <**do**> :

```
<do
  type="accept | prev | help | reset | options | delete | unknown"
label="VDATA"
name="NAME"
optional="true | false"
>
  <go> | <noop> |  <prev> | <refresh>
</do>
```

התכונה **label** מגדירה את התווית האמורה לשמש את ה-user interface widget, אולם ניתן להתעלם ממנה אם אינה יכולה להיות מוחלפת על ידי הסוכן-משתמש. מפרט WML 1.1 ממליץ כל הגבלת תווית לשישה תווים.

התכונה **name** מזהה בלעדית את קשר האירוע/משימה שהוגדר על ידי האלמנט <**do**>. שמות זהים אינם מותרים בתוך אותו כרטיס. אלמנט <**do**> ברמת הכרטיס גובר על אלמנט <**do**> בעל שם זהה שברמת החפיסה. אם לא סופק כל **name**, או אם הוא בעל מחרוזת ריקה, **name** פונה לברירת המחדל של התכונה **type**.

אם התכונה **optional** מוגדרת ל-**true**, משמעות הדבר לסוכן-משתמש היא שביכולתו להתעלם מאלמנט זה.

להלן דוגמה המראה כיצד להשתמש באלמנטי <**do**> .ex4-11.wml. היא מאפשרת לך לנווט בסדרה של כרטיסים, כולל כרטיס עזרה מקוונת לשירות מידע. חפיסה זאת מוצגת בתרשימים 4.16 עד 4.18.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
  "http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
  <card id="card1" >

<!-- the welcome card -->

    <do type="accept" label="Go!" optional="false" >
      <go href="#startcard" />
    </do>
    <do type="help" label="Help" >
      <go href="#helpcard" />
    </do>
    <p>
      Welcome to WorldFAQ, your premier source for
      international reference information.
    </p>
  </card>
```

```
<!-- the start of the actual service -->

  <card id="startcard">
    <p>
      What do you want to find:<br/>
      <a href="phones.wml"> Phone codes </a><br/>
      <a href="times.wml"> Intl times </a><br/>
      <a href="weather.wml"> Weather </a>
    </p>
  </card>

<!-- the help screen -->

  <card id="helpcard">
    <do type="accept" label="Go!" optional="false" >
      <go href="#startcard" />
    </do>
    <p>
      Select <em>Go!</em> to pick a reference service.
      You can look up country dialing codes,
      international times, and average temperatures.
    </p>
  </card>
</wml>
```

```
Welcome to WorldFAQ,
your premier source
for international
reference information

Go!              Help
```

תרשים 4.16 דוגמת <do> פשוטה

```
What do you want to
find:
▸ [ Phone codes ]
  [ Intl times ]
  [ Weather ]

Link
```

תרשים 4.17 בחירת שרות

```
Select Go! to pick a
reference service.
You can look up
country dialing
codes, international
times, and average

Go!
```

תרשים 4.18 מסך העזרה

אירועים ברמת חפיסה

כפי שאתה יכול להגדיר ברמת הכרטיס את האלמנטים <**onevent**> ו-<**do**>, ביכולתך להגדיר אותם גם ברמת החפיסה. מכיון שאירועי <**onpick**> אינם אירועים של רמת-כרטיס, אלא אירועים המופעלים על ידי אלמנטי <**option**>, אינך יכול להגדירם ברמת החפיסה.

תוכל להגדיר קשרי אירוע/משימה ברמת-חפיסה באמצעות שימוש באלמנט <**template**>. <**template**> מתאר קשרים המיוחסים לכל הכרטיסים בחפיסה. ביכולתך אז לדרוס (override) קשרים של רמת חפיסה ברמת הכרטיס. אם כרטיס מכיל קשר <**do**> בעל שם זהה לקשר <**do**> ברמת החפיסה, ניתנת העדיפות לקשר שבכרטיס. בדומה, לקשרים פנימיים ברמת הכרטיס יש עדיפות על קשרים פנימיים ברמת החפיסה, לגבי אותו סוג אירוע.

בדרך כלל תשתמש ב-<**template**> כדי להימנע מהגדרות חוזרות המיוחסות לכל הכרטיסים בחפיסה. לעיתים קרובות, קל יותר להשתמש ב-<**template**>, ואז לדרוס קשרים ייחודיים עבור כרטיסים מסוימים.

להלן התחביר של האלמנט <**template**>:

```
<template
   oneventbackward="URL"
   oneventforward="URL"
   ontimer="URL"
>
   <do>, <onevent>
</template>
```

כפי שעם תכונות כרטיס (התכונות **oneventforward**, **oneventbackward**, ו-**ontimer**) תכונות <**template**> הן צורות מצומצמות שוות ערך להגדרות <**onevent**> בעלות אורך-מלא שבגוף ה-<**template**>. אינך יכול להשתמש בקשרים שווי ערך מצומצמים ובקשרים בעלי אורך מלא באותו כרטיס או חפיסה.

להלן החפיסה הקודמת עם כרטיס עזרה מבוסס <**template**> (ex4-12.wml) :

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
 "http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
 <template>
     <do type="help" label="Help" >
       <go href="#helpcard" />
     </do>
 </template>

 <card id="card1" >

<!-- the welcome card -->

   <do type="accept" label="Go!" optional="false" >
     <go href="#startcard" />
   </do>
   <p>
     Welcome to WorldFAQ, your premier source for
     international reference information.
   </p>
 </card>

<!-- the start of the actual service -->

 <card id="startcard">
   <p>
     What do you want to find:<br/>
     <a href="phones.wml"> Phone codes </a><br/>
     <a href="times.wml"> Intl times </a><br/>
     <a href="weather.wml"> Weather </a>
   </p>
 </card>

<!-- the help screen -->

 <card id="helpcard">
   <do type="accept" label="Go!" optional="false" >
     <go href="#startcard" />
   </do>
   <p>
```

```
    Select <em>Go!</em> to pick a reference service.
    You can look up country dialing codes,
    international times, and average temperatures.
   </p>
  </card>
</wml>
```

הוספת נתונים

בדרך כלל, תוכנות יעילות יותר אם משתמש יכול להוסיף להן נתונים. אפילו אם הוספת נתונים פשוטה כבחירת פריט מתוך קבוצה, WML מספקת מיגוון מנגנונים להוספת נתונים.

הבעיה של הוספת נתונים יעילה להתקן בעל מסך קטן ויכולות מוגבלות, עדיין לא נפתרה, בין אם על ידי חומרה או על ידי עיצובי תוכנה. מקשי טלפון אינם מתאימים, זיהוי-קול פועל רק עבור יישומים בעלי פקדים ופקודות מעטים. שיטות חיזוי המנסות להעריך מה ברצונך להוסיף לפני שסיימת את פעולת ההוספה, מוגבלות על ידי חומרת הוספת הנתונים. זיהוי כתב-יד אינו מתאים מכיון שהוא דורש מסך תצוגה יקר יותר ממסך טלפון סלולרי רגיל, ולא ברור אם כדאי להשתמש בעט סימון בטלפון סלולרי. מחקר רב עדיין נדרש בשטח זה.

עד אשר יופיע מנגנון הוספת נתונים יעיל לסוג זה של התקן, עליך לנסות להקטין את כמות הוספת הנתונים הדרושה להתקן שלך. ככל שמועטת הוספת הנתונים, כן גדלה שביעות רצונו של המשתמש.

קלט משתמש

הדרך הפשוטה יותר שבה ביכולתך לקלוט נתונים לתוכנית WML היא על ידי שימוש באלמנט <**input**>. אלמנט זה מאפשר למשתמש להוסיף מחרוזת של תווים עם שליטת תבנית אפשרית המוכנסת למשתנה. כיצד יוסיפו משתמשים את הטקסט – תלוי בהתקן. הם יכולים להשתמש לדוגמה, בלוח מקשים טלפוני או בזיהוי כתב-יד. ביכולתך אז להשתמש בערך המשתנה המוסף בחפיסות וכרטיסים.

להלן התחביר של אלמנט **input**:

```
<input
  emptyok="true | false"
  format="STRING"
  maxlength="NUMBER"
  name="NAME"
  size="NUMBER"
  tabindex="NUMBER"
  title="VDATA"
  type="text | password"
  value="VDATA"
/>
```

emptyok - תכונה בוליאנית שכאשר היא מוגדרת ל-**true**, מציינת שהמשתמש אינו צריך להוסיף דבר כלשהו לאלמנט **input** זה. כרגיל, אלמנטי קלט בעלי מחרוזות **format** מחייבים את המשתמש להוסיף מחרוזת בעלת תבנית מסוימת. הגדרת ברירת המחדל ל-**emptyok** היא **false**.

format - מחרוזת המגדירה את מסכת הקלט. מחרוזת זו יכולה להכיל הן תווי בקרת מסכה והן טקסט סטטי המוצג באזור הקלט. בהמשך פרק זה נדון במפרטי תבנית תקפים.



הערה!

שדות קלט מסוימים הם תבנית שאינה יכולה להיות ריקה - לדוגמה, ארבעה מספרים ואחריהם פסיק שאחריו שמונה מספרים, או שדה קלט המכיל אותיות. אם השדה הוא אופציונלי, תרצה שיהיה ריק על ידי הגדרת התכונה emptyok ל-true, ביכולתך להגדיר שדה אופציונלי בעל תבנית ברורה כאשר השדה אינו ריק.

maxlength - המספר המירבי של תווים הניתנים להוספה לאלמנט קלט זה. ברירת המחדל היא: ללא הגבלה.

name - שם המשתנה שאליו מוקצה טקסט הקלט. אם משתנה זה כבר מוגדר ומתאים למפרט התבנית, ערכו משמש כערך ברירת המחדל לאלמנט **input**. אם **name** הוא בעל ערך שאינו מתאים למפרט התבנית, הסוכן-משתמש מנסה להשתמש בהגדרת התכונה **value** כערך ברירת המחדל. **name** היא תכונה נדרשת.

size - הרוחב בתווים של אזור קלט הטקסט במסך התצוגה.

title - מחרוזת העשויה לשמש את הסוכן-משתמש בעת הצגת האלמנט input למשתמש. הסוכן יכול להציג את title כתווית או כתיאור כלי.



הערה!

titles (כותרות) מקנים לממשק המשתמש תפקוד משופר. מספר סוכני-משתמש עוברים למצב (mode) שונה לצרכי קלט, והכותרת יכולה להזכיר למשתמש למה נכנס. שימוש אחר הוא במקרה של דפדפני-שמיעה, שיכולים להשתמש בכותרת לציון מיקוד הקלט הנוכחי.

type - אלמנט **input** שיכול להודיע למשתמש על התווים הנוספים באחת משתי דרכים. הודעת **text** המציגה את התווים בעת הוספתם. תווי **password** (סיסמה) המוצגים בצורה מעורפלת או לא קריאה כגון תו כוכבית ("*") לכל תו מוסף.

value - ערך ברירת מחדל עבור אלמנט **input** אם לא הוקצה מהמשתנה הקיים **name**. משתמשים ב-**value** אם ערכו מתאים למפרט התבנית.

tabindex - מקומו הסידורי של האלמנט בכרטיס הנוכחי. סוכני-משתמש WAP אינם חייבים ליישם את התכונה **tabindex**.

בדרך כלל, בעת שימוש באלמנטי **input**, אתה מקדם כל אלמנט בטקסט כלשהו לצרכי הדרכת המשתמש. להלן דוגמה של כרטיס הוספת נתונים פשוט (ex4-13.wml). הדפדפן מציג כל פעם שדה אחד. שדות השם הפרטי, הארץ והסיסמה מוצגות בתרשימים 4.19 עד 4.21.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
 "http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
 <card>
  <p>
   First Name:
   <input name="fname"
    maxlength="15" /><br/>

   Last Name:
   <input name="lname"
    maxlength="15" tabindex="2" /><br/>

   State:
   <input name="state"
    maxlength="2" emptyok="true"
    value="CA" tabindex="3" /> <br/>
   Zipcode:
   <input name="zipcode"
    maxlength="9" tabindex="4" /> <br/>

   Password:
   <input name="password"
    maxlength="8" type="password" tabindex="5" /> <br/>
  </p>
 </card>
</wml>
```

```
First Name:
Steve |



OK                ALPHA
```

תרשים 4.19 שדה הוספת נתונים פשוט

```
State:
CA |



OK                ALPHA
```

תרשים 4.20 שדה הוספת נתונים עם ערך מוגדר

```
Password:
****



OK                ALPHA
```

תרשים 4.21 שדה הוספת נתוני סיסמה

מפרטי תבנית

התכונה **format** של אלמנט **input** מאפשרת לך להגדיר מחרוזת בקרה לצורך ניהול יכולת המשתמש למלא שדה הוספת נתונים. כמו כן, התכונה מאפשרת לך להגדיר טקסט סטטי, כגון המקפים ("-") המפרידים בין חלקי מספר, היכולים להיות מוצגים כחלק משדה הוספת נתונים. תבנית בקרת התווים מוצגת בטבלה 4.5.

טבלה 4.5

תבנית	תווי קלט מותרים
A	סימני פיסוק או אותיות אלפבית רישיות
a	סימני פיסוק או אותיות אלפבית רגילות
N	תווי מספר
X	אותיות רישיות
x	אותיות רגילות
M	כל תו, אך הסוכן-משתמש יכול להעדיף להציג אותיות רישיות
m	כל תו, אך הסוכן-משתמש יכול להעדיף להציג אותיות רגילות
\c	הצגת "c" כתו אות יחידי בתצוגה



הערה!

באשר לתווי תבנית "M" או "m", התבנית מרמזת שסביר להניח שהתו יהיה אות רישית או רגילה בהתאמה, למרות זאת, ניתן להוסיף כל תו. השימוש ב-"M" רישית יגרום לסוכן-משתמש להתחיל באות רישית. לדוגמה, אתה עשוי לבחור עבור שם המשפחה ב-"M" ו-"m" עבור כתובת e-mail.

כאשר אתה כולל אותיות, תוך שימוש במפרט התבנית "c\", במחרוזת תבנית, הסוכן-משתמש כולל את האותיות בטקסט המוקצה למשתנה אלמנט **input**.

בנוסף, ביכולתך להשתמש בשני מפרטי תבנית אותיות-מרובות. ניתן להשתמש בהם פעם אחת בסופה של מחרוזת.

טבלה 4.6

תבנית	תווי קלט מותרים
f*	הוספת מספר כלשהו של תווים העונים על מפרט תבנית **f**
nf	הוספת מספר תווים מאחד עד תשעה כפי שהוגדר ב-**n** של פורמט **f**

להלן הדוגמה הקודמת עם מחרוזות תבנית (ex4-14.wml). תרשימים 4.22 ו- 4.23 מציגים כיצד תבנית המיקוד נראית בדפדפן. כאשר המשתמש מוסיף את התו החמישי של המיקוד, הסוכן-משתמש WAP מציג את תו המקף וממקם את עצמו לצורך הוספת הספרה הראשונה של החלק השני של המיקוד.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
  "http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
  <card>
    <p>
<!-- One uppercase letter followed by up to 14
    lowercase or punctuation characters -->

     First Name:
     <input name="fname"
       maxlength="15" format="X*M" /> <br/>

     Last Name:
     <input name="lname"
       maxlength="15" tabindex="2" format="X*M" /> <br/>
```

```
<!-- Two uppercase letters with a default value -->

      State:
      <input name="state"
        maxlength="2" emptyok="true"
        format="AA" value="CA" tabindex="3" /> <br/>

<!-- Five digits followed by up to four more digits -->

      Zipcode:
      <input name="zipcode"
        maxlength="10" tabindex="4" format="NNNNN\-*N"/> <br/>

<!-- Four to eight characters, obscured on the display -->

      Password:
      <input name="password"
        maxlength="8" type="password" tabindex="5"
        format="mmmm4m" /> <br/>
    </p>
  </card>
</wml>
```

```
Zipcode:
451|
```

תרשים 4.22 חלקו הראשון של המיקוד

```
Zipcode:
45105-|

OK
```

תרשים 4.23 חלקו השני של המיקוד

נתונים מורכבים

אין זה נדיר שכרטיס (כמו הדוגמה הקודמת) יכיל מספר אלמנטים של הוספת נתונים. ייתכן שהתקנים בעלי מסכים קטנים לא יהיו מסוגלים להציג את כל אלמנטי הקלט האלה בו-זמנית. במקרים אלה, הסוכן-משתמש חייב להחליט כיצד לטפל בממשק המשתמש. לדוגמה, האם להציג אלמנטים אלה כרשימה נגללת? אולי עדיף פריט אחד לכל דף תצוגה. כל סוכן-משתמש פותר בעיות אלה בדרכו שלו. ישנם שני רכיבי WML המספקים רמזים לסוכן-משתמש והעוזרים לו להציג את התוכן בדרך האופטימלית.

הרמז הראשון הזמין לסוכן-משתמש הוא התכונה הבוליאנית **ordered** במסגרת התגית **<card>**. כברירת מחדל, **ordered** מוגדר ל-**true**. התכונה מציינת לסוכן-משתמש שהתוכן בכרטיס מאורגן כרצף לינארי של אלמנטי שדות קלט. סוכן-משתמש יכול להציג קבוצה מסודרת של אלמנטי קלט כרצף דפים נפרדים, כשדה הוספת נתונים יחיד בכל דף, וכלי ממשק מתאים עוזר למשתמש לנווט בקלות בין הדפים. למעשה, זה מה שמתרחש בדוגמה הקודמת.

מפרט WML מציע תכונה זו כמתאימה לצורות הוספת נתונים קצרות במקום בו כל השדות דרושים. דוגמה לכך יכולה להיות הודעת e-mail הדורשת כתובת, נושא, והודעה.

אם ניקח את הדוגמה הקודמת, ונוסיף לה את התכונה **ordered** המוגדרת כ-**false**. הסוכן-משתמש מציב את כל השדות באותו מסך (ראה תרשים 4.24).

```
<wml>
  <card ordered="false">
    .
    . the same code as in the preceding example
    .
  </card>
</wml>
```

עבור תסריטים יותר מורכבים, קיים אלמנט נוסף, האלמנט <**fieldset**>, שבו תוכל להשתמש כדי לארגן אוספים של טקסט ושדות. <**fieldset**> מספק רמזים לסוכן-משתמש אודות המיקום היחסי וארגונם של אלמנטי קלט וטקסט, כך שיוכל למטב את ניווט המשתמש. אלמנט זה יעיל במיוחד להגדרת דפים לוגיים (logical pages) של תוכן ויזואלי בכרטיסים, שיכול לכלול תצוגת דפים מרובים על התקנים בעלי מסך קטן.

להלן תחביר <**fieldset**>:

```
<fieldset
   title="VDATA"
>
 <a>, <anchor>, DENTRY, <do>, FMTTEXT, <img/>, <table>, TEXT
</fieldset>
```

הסוכן-משתמש יכול להשתמש בתכונה **title** כדי להחליף את התוכן.

שים לב לשני דברים: ראשית, אלמנטים <**fieldset**> ניתנים לקינון – ביכולתך לארגן הוספת נתונים תוך שימוש בדפים-בתוך-דפים. שנית, תוכן <**fieldset**> מוגבל למספר קטן של אלמנטים מסוגים אחרים. דבר זה משפיע על תפקוד אלמנט זה כמארגן הוספת נתונים (לתיאור של אלמנט <**fieldset**>, ראה סעיף "בחירות" בהמשך הפרק).

```
1>First Name:
2 Last Name:
3 State:
4 Zipcode:
5 Password:
________________
Edit          OK
```

תרשים 4.24 כרטיס נתונים לא מאורגן

להלן דוגמה של <**fieldset**>. דוגמה זו אוספת מידע תוכן אישי, ומוסרת אותו לתסריט CGI (הקובץ ex4-15.wml). כאשר דפדפן Phone.com מגלה אלמנט <**fieldset**>, שבו תכונת הכרטיס **ordered** מוגדרת ל-**true**, הוא מציב לבדו את האלמנט <**input**> הראשון של כל <**fieldset**> בשורה הראשונה של התצוגה. הדפדפן מציב את אלמנט <**input**> הבא בשורה השנייה. כאשר המשתמש לוחץ על לחצן **OK** בסופו של ה-<**fieldset**> האחרון, מופעלת המשימה <**do**>.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
 "http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
 <card ordered="false">

<!-- cgi parm structure -->

  <do type="accept">
   <go href="piminfo.cgi" method="post">
    <postfield name="fn" value="$fname"/>
    <postfield name="ln" value="$lname"/>
    <postfield name="a1" value="$addr1"/>
    <postfield name="a2" value="$addr2"/>
    <postfield name="ct" value="$city"/>
    <postfield name="st" value="$state"/>
    <postfield name="zp" value="$zipcode"/>
    <postfield name="cn" value="$country"/>
    <postfield name="ph" value="$phone"/>
    <postfield name="fx" value="$fax"/>
    <postfield name="em" value="$email"/>
   </go>
  </do>
```

```
<!-- name information -->

    <p>
      <fieldset title="Name" >
        First Name:
        <input name="fname" maxlength="15" emptyok="false" /> <br/>
        Last Name:
        <input name="lname" maxlength="15" emptyok="false" /> <br/>
      </fieldset>

<!-- address information -->

      <fieldset title="Address" >
        Address1:
        <input name="addr1" maxlength="25" emptyok="false" /> <br/>
        Address2:
        <input name="addr2" maxlength="25" /> <br/>
        City:
        <input name="city" maxlength="20" emptyok="false" /> <br/>
        State:
        <input name="state" maxlength="15" /> <br/>
        Post Code:
        <input name="zipcode" maxlength="10" /> <br/>
      </fieldset>

<!-- contact information -->

      <fieldset title="Contact Info">
        Phone:
        <input name="phone" maxlength="25" emptyok="false" /> <br/>
        Fax:
        <input name="fax" maxlength="25" /> <br/>
        Email:
        <input name="email" maxlength="20" emptyok="false" /> <br/>
      </fieldset>
    </p>
  </card>
</wml>
```

בחירות

WML כוללת אלמנטים להגדרת רשימות בחירה, שבעזרתם יכול המשתמש לבחור פריט אחד או יותר מרשימת אפשרויות. כמו כן, תוכל ליצור רשימות בחירה מקוננות, להגדיר ערכי ברירת מחדל ולבצע משימות כאשר משתמש בוחר בפריט מסוים.

אלמנט הבחירה היסודי הוא האלמנט <**select**>. אלמנט זה מאפשר למשתמש לבחור אפשרות אחת או יותר, כאשר כל אפשרות מוגדרת על ידי האלמנט <**option**>. להלן התחביר של אלמנט <**select**>:

```
<select
  iname="NAME"
  ivalue="VDATA"
  multiple= "true | false"
  name="NAME"
  tabindex="NUMBER"
  title="VDATA"
  value="VDATA"
 >
   <optgroup>, <option>
</select>
```

לאלמנט <**select**> התכונות הבאות (הן אינן חובה):

- **iname**. שם המשתנה המגדיר את האינדקס של הבחירה או בחירות המשתמש.
- **ivalue**. האינדקס של אפשרות ברירת המחדל של הבחירה. ל-**ivalue** יש השפעה, רק אם המשתנה ששמו ניתן על ידי התכונה **iname** – לא הוגדר.
- **multiple**. תכונה בוליאנית שכאשר מוגדרת ל-**true**, מאפשרת למשתמש לבחור יותר מפריט אחד מקבוצה של אלמנטי <**option**>. הפעלת <**select**> מקצה רשימת בחירות מופרדות על ידי נקודה-פסיק ל-**name**. הגדרת ברירת המחדל היא **value**. בדרך דומה, **iname** מקצה רשימת בחירות המופרדות על ידי נקודה-פסיק. ערך ברירת המחדל הוא **ivalue**.
- **name**. שם המשתנה המגדיר את בחירת או בחירות המשתמש.
- **tabindex**. הסדר היחסי של אלמנט זה בהקשר לתוכנו של כרטיס WML זה.
- **title**. כותרת האלמנט, הניתנת לשימוש בעת הצגת האלמנט.
- **value**. ערך ברירת המחדל של המשתנה המוגדר על ידי התכונה **name**. נעשה שימוש בערך זה רק אם למשתנה לא הוגדר ערך כלשהו כאשר האלמנט מופעל לראשונה.

אלמנט <**select**> חייב לכלול אלמנט <**option**> אחד או יותר, כל אלמנט <**option**> הוא בחירה יחידה שביכולתו של המשתמש לבחור. אלמנט <**option**> יכול להכיל תוכן טקסט פשוט, הניתן להצגה.

להלן התחביר של אלמנט <**option**> :

```
<option
  onpick="URL"
  title="VDATA"
  value="VDATA"
>
 TEXT, <onevent>
</option>
```

לאלמנטי **option** התכונות הבאות :

- **onpick**. URL המופעל אם אלמנט <**option**> נבחר או אם מבוטלת בחירתו. רשימות בעלות אפשרות בחירה יחידה מפעילות את אירוע **onpick** רק כאשר פריט נבחר. רשימות מרובות-בחירה מפעילות את אירוע **onpick** כאשר פריטים יחידים נבחרים יחד או מבוטלת בחירתם.
- **title**. כותרת האפשרות. ניתן לעשות בה שימוש כאשר האפשרות מוצגת.
- **value**. הערך שנעשה בו שימוש כאשר הסוכן-משתמש מגדיר את התכונה **name** של **select**.

הדוגמה הבאה (ex4-16.wml) מיישמת צורת סקר המאפשרת למשתמשים להוסיף מידע אישי מסוים אודות עצמם (לתשומת ליבך : מסיבה כלשהי היישום לא עובד במדמה של Nokia). הכרטיס הראשון מאפשר להם לבחור <**select**> איזה חלק מהסקר ברצונם למלא, ומציג את הבחירות שבחרו לצד כל פסקת בחירה. תרשימים 4.25 עד 4.29 מציגים את התוצאה של הרצת התוכנית.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
 "http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>

 <card id="card1">
  <do type="accept" >
   <go href="piminfo2.cgi" method="post" >
    <postfield name="sex" value="$sex" />
    <postfield name="income" value="$income"/>
    <postfield name="hobbies" value="$hobbies"/>
   </go>
  </do>

<!-- starting card -->

  <p>
   Please tell us about:
   <select>
    <option onpick="#sexcard"> Your sex: $sex </option>
    <option onpick="#incomecard"> Your income: $income </option>
```

```
      <option onpick="#hobbiescard"> Your hobbies </option>
     </select>
   </p>
 </card>

<!-- sex data entry -->

 <card id="sexcard" >
   <p>
     What is your sex?
     <select name="sex" >
       <option value="Male" > Male </option>
       <option value="Female" > Female </option>
     </select>
   </p>
 </card>

<!-- income data entry -->

 <card id="incomecard" >
   <p>
     How much do you make?
     <select name="income" >
       <option value="$$10-50K" > $$10-25K </option>
       <option value="$$25-50K" > $$25-50K </option>
       <option value="$$50-100K" > $$50-100K </option>
       <option value="Over $$100K" > Over $$100K </option>
     </select>
   </p>
 </card>

<!-- hobbies data entry -->

 <card id="hobbiescard">
   <p>
     Do you have hobbies?
     <select name="hobbies" multiple="true">
       <option value="ski" > Skiing </option>
       <option value="book" > Reading </option>
       <option value="film" > Movies </option>
     </select>
   </p>
 </card>
</wml>
```

```
Please tell us about:
1>Your sex:
2 Your income:
3 Your hobbies:

OK
```

תרשים 4.25 סקר פשוט

```
What is your sex?
1>Male
2 Female

OK
```

תרשים 4.26 כרטיס המין

```
How much do you make?
1>$10-25K
2 $25-50K
3 $50-100K
4 Over $100K

OK
```

תרשים 4.27 כרטיס ההכנסה

```
Do you have hobbies?
1  Skiing
2* Reading
3*>Movies

OK
```

תרשים 4.28 כרטיס התחביבים

```
Please tell us about:
1 Your sex: Female
2 Your income:
$50-100K
3>Your hobbies

OK
```

תרשים 4.29 התוצאות הסופיות

בחירות מרובות רמה

כפי שאלמנט <**fieldset**> מאפשר לך להגדיר קשרים הירארכיים בין קבוצות של שדות הוספת נתונים, האלמנט **optgroup** מאפשר לך להגדיר קשרים הירארכיים בין אלמנטי <**option**>. תחבירו הוא :

```
<optgroup
  title="VDATA"
>
 <optgroup>, <option>
</optgroup>
```

אלמנט <**optgroup**> הוא בעל תכונה אחת יחידה – **title**. הסוכן-משתמש יכול להשתמש ב-**title** בעת הצגת קבוצת <**option**>.

קבוצת <**option**> יכולה להכיל את האלמנטים <**optgroup**> ו-<**option**>.

הדוגמה שלהלן (ex4-17.wml) משנה את כרטיס **hobbiescard** מהדוגמה הקודמת, כך שיכלול תחביבים רבים יותר המקובצים לקבוצות שונות (לתשומת ליבך: מסיבה כלשהי היישום לא עובד במדמה של Nokia).

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
 "http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>

 <card id="card1">
  <do type="accept" >
   <go href="piminfo2.cgi" method="post" >
    <postfield name="sex" value="$sex" />
    <postfield name="income" value="$income"/>
    <postfield name="hobbies" value="$hobbies"/>
   </go>
  </do>

<!-- starting card -->

  <p>
   Please tell us about:
   <select ivalue="1">
    <option value="1" onpick="#sexcard"> Your sex: $sex </option>
    <option value="2" onpick="#incomecard"> Your income: $income </option>
    <option value="3" onpick="#hobbiescard"> Your hobbies </option>
   </select>
  </p>
 </card>

<!-- sex data entry -->

 <card id="sexcard" >
  <p>
   What is your sex?
   <select name="sex" >
    <option value="Male" > Male </option>
    <option value="Female" > Female </option>
   </select>
  </p>
 </card>
```

```
<!-- income data entry -->

  <card id="incomecard" >
    <p>
      How much do you make?
      <select name="income" >
        <option value="$$10-50K" > $$10-25K </option>
        <option value="$$25-50K" > $$25-50K </option>
        <option value="$$50-100K" > $$50-100K </option>
        <option value="Over $$100K" > Over $$100K </option>
      </select>
    </p>
  </card>

<!-- hobbies data entry -->

<card id="hobbiescard">
  <p>
    Do you have hobbies?
    <select name="hobbies" multiple="true">

      <optgroup title="Sports" >
        <option value="Skiing" >    Skiing </option>
        <option value="Tennis" >    Tennis </option>
        <option value="Skydiving" > Skydiving </option>
      </optgroup>

      <optgroup title="Culture" >
        <option value="Books" > Books </option>
        <option value="Plays" > Plays </option>
        <option value="Opera" > Opera </option>
        <option value="Films" > Films </option>
      </optgroup>

      <optgroup title="Social Activities" >
        <option value="Food/Wine" > Food/Wine </option>
        <option value="Dancing" >   Dancing </option>
        <option value="Traveling" > Traveling </option>
      </optgroup>
    </select>
  </p>
</card>

</wml>
```

פרק 5

WMLScript

בנקודה מסוימת במהלך עיצוב WAP, התברר שצריך שיהיה ב-WML גם מרכיב תכנותי ליצירת משימות פשוטות, כגון בדיקת קלט משתמש ויצירת הודעות ודיאלוגים מקומיים. לצורך זה, פורום WAP הוסיף מרכיב חדש למודל יישום WAP: WMLScript. נגזרת של ECMAScript (שהיא עצמה נגזרת של JavaScript). WMLScript היא שפת תסריט קלת משקל המוסיפה יכולות רבות לסוכן-משתמש WAP. מעצבי WMLScript לקחו את התכונות הטובות והמועילות ביותר מאבותיה של השפה, אך הסירו את התכונות הבלתי נחוצות או המורכבות מדי לסביבת נתונים סלולרית.

מודל ההפעלה של WMLScript הוא מוכוון-פעולה. אין כל תוכניות ראשיות; במקום זאת, אתה יוצר קבוצת פונקציות, ומציב את קובץ הטקסט הנוצר על שרת הנגיש לתוכניות WML ו-WMLScript. כאשר פונקציה נקראת, ממיר אותה WAP gateway באופן אוטומטי לתבנית הבינארית המהודרת שלה, ומשדר נתונים אופטימליים אלה באמצעות הקשר הסלולרי. ישנם יתרונות ברורים לאחסון קוד המקור בשרת-התוכן שלך. לדוגמה, WAP gateway יכול לבצע עיבודי גרסאות עבורך – מיפויי תאימות לאחור אם השפה תשתנה בעתיד.

כדי לגשת לפונקציות ביחידת אוסף WMLScript, עליך לקרוא להן מתוכנית WML, תוך שימוש ב-URL שבו הנתיב מצביע ליחידת אוסף WMLScript והשם החלקי הוא שם הפונקציה. לדוגמה, להלן אלמנט <**go**>.

```
<go href="bships.wmls#FireTorpedo ( $ (gBoardWidth) ,
$(gBoardHeight) ) ">
```

WAP gateway מביא את קוד המקור של יחידת האוסף, מרחיב את הפונקציה **FireTorpedo** ומהדר אותה. ה-bytecodes של WMLScript מורדים (נפרקים) לסוכן-משתמש כחלק מהתגובה, ומופעל על ידי הסוכן-משתמש. כאשר הפונקציה מסתיימת, המפרש מחזיר את השליטה לכרטיס WML שקרא ל- **FireTorpedo**.

עד סיום פרק זה, אנו דנים ב- WMLScript ובספריות WMLScript הסטנדרטיות. כיון שהשפה היא בעיקרה שפה מסורתית כדוגמת C ו- Java, ומכיון שאנו מניחים שאתה יודע לתכנת, אין אנו כוללים דוגמאות קצרות רבות להדגמת תכונותיה הברורות מאליהן. לעומת זאת, אנו כוללים דוגמאות המסבירות כמה מהתכונות המורכבות יותר של השפה.

WML לעומת WMLScript

מנקודת מבטו של מתכנת, WML ו- WMLScript הם כלים נפרדים שעוצבו למשימות נפרדות. WML היא שפת סימון (declarative), המעוצבת כדי לבטא את מאפייניו של ממשק המשתמש. WMLScript היא שפת תסריט-פעולה, המעוצבת כדי ליצור מבני תכנות לוגיים וביצועיים.

ל-WAP מהלך פעולה רציף אחד בעל הקשר גלובלי מתמשך. אין בה כל מחסניות פעולה שבה פרמטרים, מצביעי פעולה ומשתנים מקומיים המוכנסים ומוצאים ממנה. היא בעלת מחסנית היסטוריה, אולם זו כוללת רק מיקומים – כרטיסים ו-URLs. WMLScript היא בעלת פונקציות, וביכולתך לקרוא לפונקציה אחת מאחרת. הביצוע מתנהל תוך שימוש במחסנית המכילה פרמטרים, מצביע ההוראות נוכחי, משתנים מקומיים ועוד. כאשר אתה קורא לפונקציה, דברים זזים במחסנית. כאשר אתה מסיים, הדברים מתחסלים ונעלמים.

כל משתני WML הם גלובליים. לעומת זאת, כל המשתנים של WMLScript הם מקומיים וקיימים כל עוד הפונקציה המכילה אותם מתבצעת. בזמן הידור, WMLScript עוקבת אחר שלושה סוגי שמות : משתנים, פונקציות ו- pragmas (אוסף מידע ברמת יחידה).

ל-WML אין משפטי הליך (למרות שמודל האירוע/משימה מספק כמה יכולות דמויות-הליך). WMLScript היא בעלת קבוצה מועילה של משפטי הליכי שפה מסורתיים, משפטי **if....else**, לולאות ועוד.

WML היא ממוקדת-תוכן וכוללת אלמנטים לעיצוב פלט וקלט. WMLScript היא ממוקדת-הליך. מלבד פונקציית ספריה אחת המאפשרת למשתמש להוסיף מחרוזת, ואחרת המאפשרת לך לרענן את תצוגת ההתקן, אין ביכולתך להוסיף נתונים ישירות. כמו כן, אין ביכולתך להציג מניפולציות בעזרת WMLScript. WML היא בעלת יכולות מוגבלות לטיפול בנתונים. ביכולתך להגדיר משתנים, ליצור רשימות בחירה וליצור מסכות-קלט להוספת נתונים. WMLScript, על אף היותה שפה חופשית (loosely typed language), היא בעלת מספר סוגי נתונים פנימיים, ספריות להמרת סוגים, מחרוזות (שביכולתך להשתמש בהן לפעולות פשוטות דמויות-מערך) ותכונות משוכללות יותר לטיפול בנתונים.

WML אינה בת-הרחבה, מלבד באמצעות שינוי מפרט WML. WMLScript כוללת תכונות להוספת ספריות פונקציה סטנדרטיות. נכון להיום, פורום WAP אינו מספק כלים או מידע למפתחי צד-שלישי, לצורך יצירת ספריות משלהם. למרות זאת, הרבה יותר קל להרחיב שפה על ידי הוספת ספריה מאשר על ידי שינוי המפרט שלה.

למרות ש-WML ו-WMLScript הם כלים נפרדים למשימות נפרדות, ביכולתך להשתמש בהם יחדיו. ניתן לקרוא לפונקציית WMLScript מתוך חפיסת WML, תוך העברת פרמטרים לפונקציה. בנוסף, ל-WMLScript יש פונקציות ספריה המאפשרות לך לקרוא ולכתוב משתני WML, ליצור הקשר חדש ולהפעיל משימות <**refresh**> מתוך תוכנית WMLScript. כמו כן, ביכולתך ליצור תור של משימות <**go**> ו- <**prev**>, לביצוע לאחר שמפרש WMLScript סיים את רצף הביצוע הנוכחי.

יסודות WMLScript

כמו כל שפות התכנות, WMLScript היא רצף של סימני-שפה רגישי גודל-אות (case-sensitive), מילים ואותיות המופרדים על ידי רווח לבן. קבוצות סימנים יוצרות משפטים, המופרדים על ידי נקודה-פסיק.

משתמשים במזהים לצורך מתן שם למשתנים, פונקציות ו- pragmas (אוסף מידע ברמת יחידה). WMLScript משתמשת לצורך מתן שמות למזהים, בכללי WML למתן שמות למשתנים.

- מזהים הנם רגישים לגודל-אות.
- הם יכולים להיות מורכבים רק מתווים אלפביתיים ומספריים ומתו קו תחתון ("_").
- הם אינם יכולים להתחיל בספרה (0..9).

ל-WMLScript יש כלל אחד נוסף לגבי מזהים: הם אינם יכולים להיות זהים לשמות שמורים. טבלה 5.1 מציגה את רשימת השמות השמורים הנוכחית של WMLScript.

WMLScript מזהה שלושה סוגי שמות: משתנים, פונקציות ו- pragmas. תוכל להשתמש באותו שם מזהה לכל אחד מסוגי השמות האלה, כך שניתן למעשה לכנות משתנה, פונקציה ופרמטר פורמלי באותו שם.

טבלה 5.1 השמות השמורים של WMLScript.

access	**domain**	**import**	**struct**
agent	**else**	**in**	**super**
break	**enum**	**invalid**	**switch**
case	**equiv**	**isvalid**	**this**
catch	**export**	**lib**	**throw**
class	**extends**	**meta**	**true**
const	**extern**	**name**	**try**
continue	**false**	**new**	**typeof**
debugger	**finally**	**null**	**url**
default	**for**	**path user**	**use**
delete	**function**	**private var**	**while**
div	**header**	**public void**	**with**
div=	**http**	**return**	
do	**if**	**sizeof**	

הערות קוד מקור באות בשתי צורות, שתיהן זהות ל-C ול-++C:

```
//single line comments

/*
multiline
comments
*/
```

סוגי נתונים

WMLScript היא שפה שאין מכריזים בה על סוגים (loosly typed langauge). אינך מכריז על סוגי נתונים, אך המהדר וסביבת ההרצה עוקבים אחר הסוג הנוכחי של משתנה – סוג הנקבע כאשר מוקצה ערך למשתנה. השפה תומכת בחמישה סוגי נתונים המוצגים בטבלה 5.2.

טבלה 5.2

Boolean	הערכים התקפים הם **true** ו-**false**
Integer	הטווח התקף הוא בין 2,147,483,648- עד 2,147,483,647. תוכל לציין מספרים הקסדצימלים ואוקטלים על ידי הוספת הקידומות "0x" או "0X" ו- "0" (אפס) בהתאמה.
Float	הטווח התקף הוא בין 1.17549435E-38-/+ עד 3.402823476E+38-/+. תוכל לציין נקודה צפה מעריכית באמצעות האותיות "e" או "E".
String	מחרוזת יכולה להתחם בין גרש או גרשיים. תוכל לכלול תווים מיוחדים במסגרת מחרוזת על ידי קידומם בקו נטוי "\". טבלה 5.3 מציגה את רשימת התווים המיוחדים.
invalid	שם זה מציין ערך לא תקף (**invalid**). ערך לא תקף יכול לקרות, לדוגמה, כאשר תוצאת פעולה של נקודה צפה היא מספר שמחוץ לטווח המותר למספרי נקודה צפה. נעשה בו שימוש כל אימת שיש צורך להבדיל פריט נתונים מארבעת סוגי הנתונים האחרים.

משתנים

כמו כל שפות התכנות, תוכניות WMLScript הן בעלות משתנים. לא מגדירים את סוגי המשתנים (loosly typed langauge) – סוג הנתונים שלהם משתנה בהתבסס על ההקצאה האחרונה אליהם. כמו כן, הם בעלי טווח הכרה מקומית (local) – קיימים כל עוד הפונקציה שבה הוכרזו מופעלת. לגושי משפטים המופרדים באמצעות סוגריים מסולסלים ("{ ו-}") אין כל השפעה על טווח ההכרה של המשתנים.

עליך להכריז על משתנים (לא על סוגם), לפני השימוש בהם בביטוי; ניתן להכריז עליהם בכל מקום שבו מתקיים משפט WMLScript. המשתנה מורכב מהמילה השמורה var ולאחריה שם רגיש לגודל-אות שיכול להכיל תווים אלפביתיים, ספרות ותו קו-תחתון. שם משתנה אינו יכול להתחיל בספרה.

טבלה 5.3 רצף תווים מיוחד

רצף	רצף תווים מיוחדים
\'	גרש
\"	גרשיים
\\	לוכסן שמאלי
\/	לוכסן ימני
\b	רווח אחורה
\f	קידום טופס
\n	שורה חדשה
\r	החזרת גררה
\t	תו אופקי
\xhh	תו ISO8859 לקידוד הספרות ההקסדצימליות hh
\ooo	תו ISO8859 לקידוד הספרות האוקטליות ooo
\uhhhh	תו Unicode לקידוד הספרות ההקסדצימליות hhhh

הכרזה יכולה להכיל ערך אתחול אופציונלי. אם קיים כזה, מהדר WMLScript משתמש בערך זה כדי לקבוע את סוג האתחול של המשתנה.

אופרטורים

מכיון ש-WMLScript מעוצבת לאספקת הליכים לוגיים, היא מכילה קבוצה עשירה של אופרטורים לוגיים ומספריים למשימות. אתה משלב משתנים ושמות משתנים ואופרטורים, כדי ליצור ביטויים המוערכים בסדר מסוים. סדר הערכת ביטוי תלוי באופרטור המקדם והקשרו האסוציאטיבי.

טבלה 5.4 מציגה:

- קבוצת האופרטורים השלמה של WMLScript.
- ברירת מחדל של עדיפויות החישוב של אופרטורים בביטוי (ניתן לבטל העדפות אלה באמצעות סוגריים).

- הסדר בו אופרנדים (מופעלים) מחושבים - משמאל לימין ("L") או מימין לשמאל ("R") ;
- סוגי אופרנדים תקפים וסוג ערך התוצאה.
- תיאור קצר.

להלן מספר הערות אודות האופרטורים הפחות ידועים :

typeof מחזיר integer המתאר את סוגו הפנימי של האופרנד.

טווח הערך התקף הוא מאפס לארבע, ומייצג בהתאמה integer, נקודה צפה, מחרוזת, ערך בוליאני ו- **invalid.**

isvalid מחזיר ערך בוליאני המציין את הסוג התקף של האופרנד שלו. false מציין שהאופרנד אינו ביטוי תקף. לדוגמה, הערך המוחזר של **isvalid(1/0)** הוא **false.**

האופרטור המותנה של WMLScript הוא אופרטור WMLScript המשולש היחידי. הוא מכיל שלושה אופרנדים. אם האופרנד הראשון מוערך ל-true הוא מחזיר את תוצאת החישוב של האופרנד השני; אם לא, הוא מחזיר את תוצאת החישוב של האופרנד השלילי.

```
result = male ? "Jhon" : "Mary";
```

אם חישוב **male** הוא **true**, התוצאה מוגדרת כ-"John"; אם לא, התוצאה מוגדרת כ- "Mary".

אופרטור הפסיק (","), שהוא האופרטור בעל העדיפות הנמוכה ביותר, מאפשר לך לחשב ביטויים מרובים במקום ביטוי יחיד. תוכל להשתמש בו, לדוגמה, כדי לאתחל יותר ממשתנה אחד במסגרת לולאת אתחול **for.** אופרטור הפסיק מחזיר את תוצאת האופרנד האחרון שלו.



הערה!

רשימת פרמטרים בקריאות פונקציה אינן מופרדות על ידי האופרטור פסיק, פשוט על ידי פסיקים פשוטים. אם ברצונך לכלול את אופרטור הפסיק כחלק מפרמטר המועבר לפונקציה, עליך להקיף אותו בסוגריים.

המרות סוג

WMLScript מבצעת כל המרת סוג שביכולתה לבצע. אם ההמרה נראית הגיונית, סביר שהיא בוצעה על ידה. ישנם מספר מקרים מיוחדים שעליך לדעת אודותם :

- **true** בוליאני מומר למחרוזת "true". שלם (integer) שאינו אפס ומספרי נקודה-צפה מומרים ל-**true** בוליאני.
- **false** בוליאני מומר למחרוזת "false". שלם 0 ומספר נקודה צפה 0.0 מומרים ל-**false** בוליאני.

- המחרוזת הריקה (" ") מומרת ל-**false** בוליאני. כל יתר המחרוזות מומרות ל-**true**.
- המחרוזת הריקה אינה ניתנת להמרה לשלם (integer) או למספר נקודה צפה.



הערה!

רוב האופרטורים של WMLScript, אלה עם דגל התוצאה כוכבית ("") שבטבלה 5.4 מחזירים ערך לא תקף אם המרת הנתונים המשתמעת שלהם (implicit data conversions) נכשלה, או אם אחד האופרנדים אינו תקף.*

פונקציות

לצורך שימוש ב-WMLScript, עליך ליצור אוסף של פונקציות, ולהציב אותן ביחידת הוספה, קובץ נפרד של קוד-מקור מהודר. ליחידת ההוספה אין משתנים גלובליים ולא תוכנית מרכזית. ניתן להשוותה לספריית שגרות (subroutine).

כל פונקציה מוכרזת בעזרת מילת המפתח **function**, ואחריה רשימת פרמטרים אופציונלית, גוש משפטים המכיל את גוף הפונקציה. ישנם מספר כללים שאסור לחרוג מהם כשמגדירים פונקציות:

- שמות הפונקציה חייבים להיות ייחודיים במסגרת יחידת ההוספה.
- הכרזות פונקציה אינן ניתנות לקינון.
- כל הפרמטרים מועברים לפי ערך (by value) - הם מתפקדים כמשתנים מקומיים שאותחלו לפני ביצוע גוף הפונקציה.
- קריאות פונקציה חייבות להכיל מספר זהה של פרמטרים למספרם בעת הכרזת הפונקציה.
- פונקציות תמיד מחזירות ערך. ערך ברירת המחדל הוא מחרוזת ריקה.

טבלה 5.4 האופרטורים של WMLScript

Precedence[13]	Associativity	Operator	Operand types	Result type	Operation Performed
1	R	++	number	number*	pre- or post-increment (unary)
1	R	--	number	number*	pre- or post-decrement (unary)
1	R	+	number	number*	unary plus
1	R	-	number	number*	unary minus (negation)

Precedence[13]	Associativity	Operator	Operand types	Result type	Operation Performed
1	R	~	integer	integer*	bitwise NOT (unary)
1	R	!	boolean	boolean*	logical NOT (unary)
1	R	typeof	any	integer	return internal data type (unary)
1	R	isvalid	any	boolean	check for validity (unary)
2	L	*	numbers	number*	multiplication
2	L	/	numbers	floating-point	division
2	L	div	integers	integer*	integer division
2	L	%	integers	integer*	remainder
3	L	-	numbers	number*	subtraction
3	L	+	numbers or strings	number or string*	addition (numbers) or string concatenation
4	L	<<	integers	integer*	bitwise left shift
4	L	>>	integers	integer*	bitwise right shift with sign
4	L	>>>	integers	integer*	bitwise right shift with zero fill
5	L	<, <=	numbers or strings	boolean*	less than, less than or equal
5	L	>, >=	numbers or strings	boolean*	greater than, greater or equal
6	L	==	numbers or strings	boolean*	equal (identical values)
6	L	!=	numbers or strings	boolean*	not equal (different values)
7	L	&	integers	integer*	bitwise AND
8	L	^	integers	integer*	bitwise XOR
9	L	|	integers	integer*	bitwise OR

Precedence[13]	Associativity	Operator	Operand types	Result type	Operation Performed
10	L	&&	booleans	boolean*	logical AND
11	L	||	booleans	boolean*	logical OR
12	R	? :	boolean, any, any	any*	conditional expression
13	R	=	variable, any	any	assignment
13	R	*=, -=	variable, number	number*	assignment with numeric operation
13	R	/=	variable, number	floating-point	assignment with numeric operation
13	R	%=, div=	variable, integer	integer*	assignment with integer operation
13	R	+=	variable, number or string	number or string*	assignment with addition or concatenation
13	R	<<=, >>=, >>>=, &=, ^=, |=	variable, integer	integer*	assignment with bitwise operation
14	L	,	any	any	multiple evaluation

להלן התחביר הבסיסי להכרזת פונקציות (פריטים בסוגריים מרובעים הם אופציונליים):

```
[extern] function funcname { [parameter list] }
{
// function body
};
```

השתמש במילת המפתח **extern** כדי לציין פונקציה הניתנת לקריאה מחוץ ליחידת ההוספה שלה.

ישנן שלוש שיטות באמצעותן ניתן לקרוא לפונקציות WMLScript מפונקציות אחרות, תלוי במיקומה של הפונקציה הקוראת. השיטה הראשונה, המוכרת ביותר, היא קריאה לפונקציה מתוך אותה יחידת הוספה. יש לכלול את שם הפונקציה והפרמטרים שלה בביטוי:

```
x = foo_function ( parm1, parm2 );
```

השיטה השנייה משמשת לקריאה לפונקציות מיחידות הוספה אחרות. יש להתחיל עם משפט הקידומת **use url** (ראה סעיף "יחידות הוספה" בפרק זה לפירוט נוסף אודות pragmas של WMLScript):

```
use url unitName unitUrl
```

כגון:

```
use url FooUnit "http://www.worldfaq.com/foo";
```

אז ניתן לקרוא לפונקציה באותו אופן בו מתייחסים לחפיסת WML, תוך שימוש בעוגן החלקי ורשימת פרמטרים:

```
x = FooUnit#foo_function (parm1, parm2);
```

כדי לאפשר לפונקציה נקראת לפעול, יש להכריז על foo_function כ- **extern** ביחידת ההשלמה **FooUnit**, והפונקציה חייבת להיות בעלת שני פרמטרים בדיוק.

השיטה השלישית לקריאת פונקציית WMLScript היא עבור פונקציות הנכללות בספריות הסטנדרטיות של WMLScript (לפרטים אודות הספריות הנוכחיות, ראה סעיף "ספריות WMLScript"). כל שיש לעשות הוא להקדים את שם הפונקציה לשם הספריה. לדוגמה, כדי להשתמש במספר הנקודה הצפה הגדול ביותר המותר, יש להשתמש ב:

```
var maxfloat =Float.max ();
```

משפטים

WMLScript היא בעלת קבוצה שימושית של משפטי הליך הניתנים לשימוש. ניתן לסווג לשלושה סוגים: משפטים פשוטים, משפטי בקרת זרימה ומשפטי עצירת זרימה.

כבר ראית דוגמאות של רוב המשפטים הפשוטים של WMLScript:

```
;                              // המשפט הריק
var maxfloat, minfloat         //var משפט
    maxfloat =Float.Max ();    // משפט הביטוי
{ };                           // משפט הבלוק
```

ישנם שלושה משפטי בקרת זרימה שפעולתם שוות ערך למשפטים זהים ב-C:

```
if ( expression ) statement [else statement]
while ( expression ) statement
for ( [expression] ; [expression] ; [expression] ) statement
```

עבור כל הביטויים המחושבים לערך בוליאני, **invalid** נחשב ל-**false**.

לסיום, ישנם שלושה משפטי עצירת זרימה שגם פעולתם שוות ערך למשפטים זהים ב-C: **break**, **continue** ו-**return**. **Break** מפסיק את מעגל לולאת **for** או **while** הנוכחית. **continue** מפסיק את מעגל לולאת **for** או **while** הנוכחית, אך אינו מפסיק את הלולאה עצמה. **return** יוצא מהפונקציה הנוכחית ויכול להחזיר ערך.

יחידות הוספה

מבנה שפת WMLScript האחרון שעליך לדעת הוא יחידת ההוספה. יחידת הוספה היא קובץ קוד מקור המהודר בנפרד ומכיל הגדרות של קבוצת פונקציות קשורות.

יחידות הוספה יכולות להכיל pragmas - משפטים המציינים מידע ברמת יחידת ההוספה. pragmas מתחילים כולם במילת המפתח **use**, רובם מופיעים בתחילתה של יחידת ההוספה, לפני כל הכרזה על פונקציה. ל-WMLScript יש שלושה pragmas: **url**, **access** ו-**meta**.

url pragma

כבר ראית את ה-pragma **url** בדוגמה קודמת. תשתמש בה לצורך התייחסות לפונקציות שביחידות הוספה אחרות:

```
use url FooUnit "http://www.worldfaq.com/foo";

 function UseFoo () {
   var  x = FooUnit#foo_function (5, 6);
}
```

אם יש לך pragmas **url** רבים, כולם חייבים להימצא בתחילת קובץ המקור.

access pragma

ה-pragma השנייה היא ה- pragma **access**. היא מגדירה מספר כללים, תוך שימוש ב- domain name והגדרות נתיב, הקובעות למי תהיה גישה לפונקציה של יחידת הוספה. קריאות פונקציה חייבות להיקרא מ-URLs התואמים להגדרות ה-**domain** ו-**path**.

ההתאמה חייבת להיות לשמותיהם המלאים של ה-domain וה-path. התאמות תו כללי (wildcard) ייחודי אינן אפשריות. לדוגמה, "wapforum.org" תואמת ל-domain בשם "org", אך לא ל-domain בשם "forum.org". ברירת המחדל של ה-domain היא ה-domain של יחידת ההוספה. ברירת המחדל של הנתיב היא "/".

נתיבים יכולים להיות יחסיים או מוחלטים. נתיבים יחסיים מומרים לנתיבים מוחלטים על ידי הסוכן-משתמש לפני שהם נבדקים מול ה-pragma **access** של יחידת ההוספה.

להלן התחביר של pragma **access**. ישנן שלוש צורות המוצגות כשמפריד ביניהן קו אנכי ("|"):

```
use access
 domain  domain_name |
 path    path_name    |
 domain  domain_name  path  path_name ;
```

ליחידת ההוספה יכולה להיות רק pragma **access** יחידה. כברירת מחדל שליטת access אינה זמינה.

לשליטת ה-access הבאה:

```
use access domain "worldfaq.com" path "/stats" ;
```

ה-URLs הבאים יכולים לגשת לפונקציות **extern** שביחידת ההוספה:

```
http://www.worldfaq.com/stats/run1
http://www1.worldfaq.com/stats/bin-cgi/example1
```

ה-URLs הבאים אינם יכולים לגשת לפונקציית **extern** כלשהי שביחידת ההוספה:

```
http://www.worldfaq.com/run1
http://www1.worldfaq.com/bin-cgi/example1
```

meta pragma

הסוג האחרון של pragma הוא ה-pragma **meta**. השתמש בה לצורך הגדרת מידע העשוי להועיל לישות בסביבת יישום WAP.

עליך להגדיר meta-information - מידע עם שם אפיון ואחריו ערך האפיון. בנוסף לכך, meta-information מסוים יכול להכיל שם סכמה לצורך פירוש הנתונים. להלן התחביר של pragma **meta**:

```
use meta
  name        propName "propvalue";  |
  http equiv   propname "propValue"; |
  user agent  propName "propValue";
```

ישנם שלושה סוגי meta-information: name, http equiv ו-user agent.

meta-information מסוג **name** אמור לשמש שרתי מקור:

```
use meta name "copyright"  "(c) Phone.com , 2001"
```

סוכני-משתמש אמורים להתעלם ממידע זה.

meta-information מסוג **http equiv** אמור להיות מפורש על ידי שרת WAP ככותרות HTTP (למידע נוסף אודות כותרות HTTP, ראה פרק 8):

```
use meta http equiv "Keywords" "Reference, Encyclopedia";
use meta http equiv "Cache-control" "no-cache";
```

הם מומרים לכותרות תגובה של WSP או HTTP על ידי WAP gateway אם יחידת ההוספה הודרה לפני שהגיעה לסוכן-משתמש.

meta-information מסוג **user agent** מכוון לסוכני-משתמש.

```
Use meta user agent "persistent_store" "X:234, Y:122,  Z:672"
"Pairs";
```

הוא חייב להימסר לסוכן-משתמש, ולא להיות מוסר על ידי רשת מתווכת כלשהי. נכון לעכשיו, תכונה זו אינה בשימושם של סוכני-משתמש תואמי-WAP כלשהם. ניתן להשתמש בתכונה באמצעות דפדפנים מסוימים, לצורך יישומן של הרחבות WAP.

ספריות WMLScript

WMLScript חסרה יכולות שפת תכנות בסיסיות, כגון ניהול מחרוזות, פונקציות אריתמטיות חזקות וממשק עבור תוכניות WML 1.1, בהיותה מעוצבת להיות שפה רזה ויעילה (lean and mean). למרבה המזל, WMLScript 1.1 היא בעלת שש ספריות סטנדרטיות המוסיפות לשפה יכולת תפקוד לא מעטה: **Lang**, **Float**, **String**, **URL**, **WMLBrowser** ו-**Dialogs**.

הספריות **Float**, **String** ו-**URL** מכילות בהתאמה, פונקציות לטיפול במספרי נקודה צפה, מחרוזות ו-URLs. הספריה **Dialogs** מספקת לך פונקציות לצורך הצגת דיאלוגים למשתמש. הספריה **Lang** מתגברת את WMLScript בפונקציות אריתמטיות נוספות. הספריה **URL** מאפשרת לך להשיג רכיבי URL ולנהל כתובות URL. הספריה **WMLBrowser** מספקת ממשק פשוט לקריאה וכתיבה של משתני WML ולהרחבתן של משימות WML מסוימות.

סוכן-משתמש תואם WML 1.1 חייב לתמוך בכל הספריות האלה מלבד בספריית **Float**. עבור התקני שלם-בלבד החסרים ספריית **Float** קיימות מספר אזהרות שעליך לשים לב אליהן:

- **Lang.float**, הפונקציה המשמשת לבדיקה האם קיימת תמיכה בנקודה צפה, מחזירה **False**.
- פונקציות ספריה מקבלות ארגומנטים רק מסוג בוליאני, שלם, מחרוזת ו-**invalid**.
- התעלמות מכל כללי המרת נקודה צפה.
- הפונקציה **Lang.parseFloat** מחזירה **invalid**.
- כל הפונקציות של הספריה **Float** מחזירות **invalid** כאשר מתרחשת שגיאה בהפעלתן.

ספריות WMLScript משתמשות בטכניקות פשוטות של טיפול-בשגיאה. אם פרמטר כלשהו הוא **invalid**, הפונקציה מחזירה **invalid**. כמו כן, אם פרמטר אינו מהסוג הצפוי על ידי הפונקציה ואינו ניתן להמרה לסוג הנכון, הפונקציה תחזיר **invalid**.

בהמשך פרק זה מצוין שמה של כל פונקציה ואחריו רשימת הפרמטרים. כל פרמטר מציין את סוג הנתונים הדרוש לו. הספריות משתמשות בכללי המרת סוגי נתונים סטנדרטיים של WMLScript. כל שוני מכללי ברירת מחדל אלה יפורט בתיאורי הפרמטרים.

כל הפרמטרים של הפונקציות מועברים לפי ערך (by value). כאשר פונקציה כדוגמת הפונקציה לטיפול במחרוזות מחזירה ערך, זוהי ישות נתונים חדשה. אין כל שינוי בערכי הפרמטרים המקוריים.

נספח ד' כולל סיכום של כל פונקציות WMLScript, מקובצות בסדר שבו הן מוצגות בסעיפים הבאים.

הספריה Lang

הספריה **Lang** מספקת מספר יכולות ליבת-שפה. ניתן למיין את הפונקציות שלה לפונקציות אריתמטיות, המרה, בקרת זרימה, סביבה ומספר אקראי.

פונקציות אריתמטיות

בספריית **Lang** קיימות שלוש מהפונקציות האריתמטיות הבסיסיות. לספריית **Float** יש יכולות אריתמטיות מורכבות יותר.

- **abs** (*number*). מחזירה את הערך המוחלט של *number*, באותו סוג של *number*.
- **max** (*number1,number2*). מחזירה את המספר היותר גדול מבין שניהם ובסוג המספר הנבחר. אם המספרים שווים, מוחזר המספר הראשון.
- **min** (*number1,number2*). מחזירה את המספר היותר קטן מבין שניהם ובסוג המספר הנבחר. אם המספרים שווים, מוחזר המספר הראשון.

פונקציות המרה

פונקציות ההמרה של הספריה **Lang** מאפשרות לך לבדוק המרת מחרוזות למספרים שלמים ומספרי נקודה צפה.

- **isFloat** (*value*). מחזירה **true** אם ניתן להמיר את *value* למספר נקודה צפה, תוך שימוש ב- **parseFloat()**; אם לא, מוחזר **false**.

```
var  x  = Lang.isFloat  (" 135E-4, a real #");    // x = true
var  x  = Lang.isFloat  ("+125E");                // x = false
```

- **isInt** (*value*). מחזירה **true** אם ניתן להמיר את *value* לשלם, תוך שימוש ב-**parseInt**; אם לא, מוחזר **false**.

```
var  x  = Lang.isInt  ("135.25");    // x = true
var  x  = Lang.isInt  ("temp");      // x = false
```

- **parseFloat** (*string*). מחזירה את מספר הנקודה הצפה השווה ל- *string*. הניתוח פוסק בתו הראשון שאינו חלק ממספר נקודה צפה תקף. **ParseFloat()** מחזירה **invalid** אם אירעה שגיאת ניתוח או אם ההתקן אינו תומך במספרי נקודה צפה.

```
var  x  = Lang.parseFloat  ("135E-4, a real #");    // x = 0.0135
var  x  = Lang.parseFloat  ("+125E");               // x = invalid
```

- (*string*) **parseInt**. מחזירה שלם השווה ל- *string*. הניתוח פוסק בתו הראשון שאינו חלק משלם תקף. **ParseInt()** מחזירה **invalid** אם אירעה שגיאת ניתוח.

```
var  x  = Lang.parseInt  ("135.25");    // x = 135
var  x  = Lang.parseInt  ("temp");      // x = invalid
```

פונקציות סביבה

פונקציות הסביבה של Lang מאפשרות לך לחקור את הסוכן-משתמש אודות יכולותיו.

- **characterSet()**. מחזירה שלם שהוא הערך שהוקצה על ידי IANA, המזהה את קבוצת התווים הנתמכת על ידי מפרש WMLScript.
- **float()**. מחזירה **true** אם קיימת תמיכה במספרי נקודה צפה. אם לא – מוחזר **false**.
- **maxInt()**. מחזירה את ערך השלם המקסימלי (2,147,483,647).
- **minInt()**. מחזירה את ערך השלם המינימלי (-2,147,483,648).

פונקציות ניהול זרימה

פונקציות ניהול הזרימה של **Lang** מאפשרות לך להפסיק את פעולת הביצוע הנוכחית של WMLScript, ומחזירות את השליטה לישות הקוראת.

- (*string*) **abort**. מפסיקה את הפירוש הלא רגיל (interpretation abnormally) של WMLScript bytecode הנוכחי, ומחזירה מחרוזת המתארת את השגיאה. מחרוזת זו אינה נגישה במסמך WML.
- (*value*) **exit**. מפסיקה את הביצוע, מחזירה *value* ליישות הקוראת. מחרוזת זו אינה נגישה במסמך WML.

פונקציות מספר אקראי

ספריית **Lang** מכילה מחולל (פונקציה ליצירה של) מספרים שלמים אקראיים.

- (*integer*) **random**. מחזירה מספר אקראי שלם חיובי בין אפס ל-*integer*. **random** מחזירה **invalid** אם השלם קטן מאפס.
- (*integer*) **seed**. מאתחלת את מחולל המספר האקראי, ומחזירה מחרוזת ריקה. אם ערך **seed** הוא מספר נקודה צפה, **seed** משתמשת ב-**Float.int**, אם היא זמינה. אם **Float.int** אינה זמינה, **seed** תחזיר **invalid**.

הספריה Float

הספריה **Float** מכילה שתי פונקציות סביבה ושש פונקציות אריתמטיות. אם פעולות נקודה צפה אינן נתמכות בהתקן מסוים, כל הפונקציות של הספריה **Float** יחזירו **invalid**. תוכל לקרוא ל-**Lang.float** כדי לדעת אם פעולות הנקודה הצפה זמינות.

פונקציות סביבה

פונקציות הסביבה של **Float** מאפשרות לך לדעת את טווח מספרי הנקודה הצפה.

- **maxFloat**( ) - מחזירה את ערך הנקודה הצפה המקסימלי (3.40282347E+38).
- **minFloat**( ) - מחזירה את ערך הנקודה הצפה המינימלי (1.17549435E-38).

פונקציות אריתמטיות

הפונקציות האריתמטיות של **Float** מספקות יכולות אריתמטיות מעבר לקיים בספריה **Lang**.

- **ceil** (*number*). מחזירה את ערך השלם היותר קטן הנמוך מ-*number*.

```
var x = Float.ceil  ( "2.5" );     // x = 3
var x = Float.ceil  ( "-2.5" );    // x = 2
```

- **floor** (*number*). מחזירה את ערך השלם היותר גדול, שאינו הגבוה מ-*number*.

```
var x = Float.floor  ( "2.5" );     // x = 2
var x = Float.floor  ( "-2.5" );    // x = -3
```

- **int** (*number*). מחזירה את חלק השלם של *number*.
- **pow** (*number1, number2*). מחזירה את *number1* בחזקת *number2*. אם *number1* שלילי, *number2* חייב להיות שלם. אם *number1* הוא אפס ו- *number2* קטן מאפס, **pow** תחזיר **invalid**.

```
var x = Float.pow  ( 3, 2.0 );              // x = 9
var x = Float.pow  ( -1, 2.0 );             // x = 1
var x = Float.pow  ( -1, 1.5 );             // x = invalid
var x = Float.pow  ( false , "-1.5" );      // x = invalid
```

- **round** (*number*). מחזירה את השלם היותר קרוב ל-*number*. אם שני שלמים קרובים באותה מידה ל-*number*, **round** תחזיר את הגדול שביניהם.

```
var x = Float.round ( 2.5 );         // x = 3
var x = Float.round ( "-2.5" );    // x = -2
```

- **sqrt** (*number*). מחזירה את השורש הריבועי של *number*, **sqrt** תחזיר **invalid** אם *number* קטן מאפס.

הספריה String

WMLScript היא בעלת ספריה רבת עוצמה לטיפול במחרוזות. הפונקציות הכלולות בה ניתנות לסיווג לבסיסיות, תת-מחרוזות ופונקציות המרה. פונקציות האלמנט מחלקות מחרוזות לקבוצה של תת-מחרוזות, המופרדות על ידי תו ייחודי, ומאפשרות לך לפעול על אלמנטים יחידים שבמחרוזת. כל פעולות המחרוזת הן רגישות לגודל אות (case sensitive).

פונקציות בסיסיות

הפונקציות הבסיסיות מאפשרות לך לבחון אורך מחרוזת, להרחיב תווים ייחודיים ולהסיר רווחים לבנים ממחרוזת.

- **charAt** (*string, number*). מחזירה מחרוזת בת תו יחיד מ-*string*, המכילה את התו במיקום *number*. אם *number* הוא מספר נקודה צפה, הוא מומר תחילה לשלם בעזרת **Float.int** אם היא זמינה; אם לא, מוחזר **invalid**. אם *number* הוא מחוץ לטווח, **charAt** תחזיר מחרוזת ריקה. מפתח התווים מתחיל מאפס.
- **compare** (*string1, string2*). מבצעת השוואה לשונית בין *string1* ל- *string2*, תוך שימוש בקודי התו של קבוצת התווים המקומית של הסוכן-משתמש. **compare** מחזירה מינוס אחד אם *string1* קטנה מ- *string2*, 0 אם הן זהות ו- 1 אם *string2* קטנה מ-*string1*. קבוצת התווים המקומית היא הקבוצה המזוהה על ידי **Lang.characterSet**.
- **isEmpty** *(string)*. מחזירה **true** אם אורך *string* הוא 0; אם לא, **false**.
- **length** (*string*). מחזירה שלם באורך *string*.
- **squeeze** (*string*) - מחזירה מחרוזת השווה ל- *string*, כאשר כל הרווחים הלבנים הרציפים הוסרו. תווי רווח לבן כוללים טאבים אופקיים ואנכיים, קידום דף, רווחים, קידום שורה והחזרת גררה.
- **trim** (*string*) - מחזירה מחרוזת השווה ל- *string* כאשר כל הרווחים שבראש המחרוזת ובסופה הוסרו. תווי רווח לבן כוללים טאבים אופקיים ואנכיים, קידום דף, רווחים, קידום שורה והחזרת גררה.

פונקציות תת-מחרוזת

פונקציות תת-מחרוזת עוסקות בזיהוי ובפעולה על מחרוזת המהווה חלק ממחרוזת גדולה יותר. אינדקס תת-מחרוזות מתחיל באפס.

- **subString** (*string*, *startNumber*, *lengthNumber*). מחזירה תת-מחרוזת של מחרוזת המתחילה במיקום startNumber ואורכה הוא lengthNumber. אם startNumber קטן מאפס, אזי אפס משמש כמיקום ההתחלה. אם lengthNumber גדול ממספר התווים במחרוזת המתחילים מ-startNumber, התווים הנותרים

מוחזרים. אם startNumber גדול מאורך המחרוזת או אם lengthNumber קטן או שווה לאפס, **subString** תחזיר מחרוזת ריקה.

```
var  x = String.subString ( "Hello Dolly!", 0, 5);        // x = "Hello"
var  x = String.subString ( "Hello Dolly!", -1, 5);       // x = "Hello"
var  x = String.subString ( "Hello Dolly!", 6, 10);     // x = "Dolly!"
var  x = String.subString ( "Hello Dolly!", 15, 0);    // x = " "
```

- **find** (*string*, *subString*). מחזירה את השלם המציין את המיקום הראשון במחרוזת המתאים לתת המחרוזת. אם אין התאמה, **find** תחזיר 1-. התאמה מחייבת זהות בייצוג התווים.

```
var  x = String.find ("Hello Dolly!", "doll" );   // x = 6
var  x = String.find ("Hello Dolly!", "dog" );  // x = -1
```

- **replace** (*string*, *oldString*, *newString*). מחזירה מחרוזת חדשה, שבה כל המופעים של oldString מוחלפים ב-newString. התאמה מחייבת זהות בייצוג התווים.

```
var  x = String.replace ("Hello Dolly!", "doll", "dog" );      // x ="Hello Dolly!"
var  x = String.replace ("Hello Dolly!", "Doll", "dogg" );  // x ="Hello doggy!"
```

פונקציות אלמנט

פונקציות אלמנט סורקות מחרוזת אחר אלמנט אחד או יותר – תת-מחרוזות המופרדות על ידי תו החוזר על עצמו (recurring). המחרוזת הריקה נחשבת כאלמנט תקף. למרות שניתן להגדיר את התו המפריד באמצעות מחרוזת שיכולה להיות בעלת אורך גדול מ- 1, רק התו הראשון משמש לזיהוי אלמנטים. ספירת האלמנטים מתחילה מאפס.

- **elementAt** (*string*, *number*, *sepString*). מחזירה את האלמנט מספר *number* במחרוזת, שבה אלמנט הוא כל תת-מחרוזת, כולל מחרוזת ריקה שמופרדת על ידי התו הראשון של *sepString*. התו הראשון באלמנט מתחיל במקום אפס.

  אם *number* קטן מאפס, האלמנט הראשון מוחזר. אם הוא גדול ממספר האלמנטים שבמחרוזת, האלמנט האחרון מוחזר. אם המחרוזת היא המחרוזת הריקה, המחרוזת הריקה מוחזרת. אם המספר הוא מספר נקודה צפה, תחילה הוא מומר לשלם בעזרת **Float.int** אם היא זמינה; אם לא, מוחזר **invaild**. **elementAt** מחזירה **invalid** אם *sepString* היא מחרוזת ריקה.

```
var  x = String.elementAt ("Hello Dolly! ", 0, " # " );      // x = "Hello"
var  x = String.elementAt ("Hello Dolly! ", 1, " ! " );       // x = "Dolly!"
var  x = String.elementAt ("Hello Dolly! ", 2, "  " );       // x = " "
var  x = String.elementAt ("Hello Dolly! ", 3, " # " );    // x = " "
var  x = String.elementAt ("Hello Dolly! ", 0, " " );      // x = invalid
```

- (*string*, *sepString*) **elements**. מחזירה שלם המונה את מספר התת-מחרוזות ב- *string* כולל מחרוזות ריקות, המופרדות על ידי התו הראשון של *sepString*. **elements** מחזירה **invalid** אם *sepString* היא מחרוזת ריקה.

```
var  x =  String.elements ("Hello Dolly !",  "  " );      // x = 3
var  x =  String.elements (" Hello Dolly ! ",  ",  " );  // x = 5
var  x =  String.elements ("Hello Dolly !",  "; " );    // x = 0
var  x =  String.elements ("Hello Dolly!",  " " );     // x = invalid
```

- (*string*, *elemString, number, sepString*) **insertAt**. מחזירה מחרוזת חדשה המורכבת מ- *string* ומ- *elemString* (ומ- *sepString* אם דרוש), המוספת כאלמנט מספר *number*. אינדקס האלמנט מתחיל באפס.

אם *number* קטן מאפס, *elemString* מוספת כאלמנט הראשון. אם *number* גדול ממספר האלמנטים במחרוזת, *elemString* מוספת כאלמנט האחרון. אם *string* היא מחרוזת ריקה, מחרוזת חדשה שהוגדרה כ-*elemString* מוחזרת. אם *number* הוא מספר נקודה צפה, הוא מומר תחילה לשלם בעזרת **Float.int** אם היא זמינה; אם לא, מוחזר **invalid**. **insertAt** מחזירה **invalid** אם *sepString* היא מחרוזת ריקה.

```
var x = String.insertAt
  (  "x=2, y=3, z=4",  "t=16",  0,  "," );   //  x = "t=16, x=2 ,y=3, z=4"
var x = String.insertAt
  (  "x=2, y=3, z=4",  "t=16",  5,  "," );   //   x = "x=2 ,y=3, z=4, t=16"
```

- (*string, number, sepString*) **removeAt**. מחזירה מחרוזת חדשה, שבה אלמנט מספר *number* מוסר. אלמנט הוא כל תת מחרוזת של *string* המופרד על ידי התו הראשון של *sepString*. מפתח האלמנט מתחיל מאפס.

אם *number* קטן מאפס, האלמנט הראשון מוסר. אם הוא גדול ממספר האלמנטים ב-*string*, האלמנט האחרון מוסר. אם *string* היא מחרוזת ריקה, המחרוזת הריקה מוחזרת. אם *number* הוא מספר נקודה צפה, תחילה הוא מומר לשלם בעזרת **Float.int** אם היא זמינה; אם לא, מוחזר **invalid**. **removeAt** מחזירה **invalid** אם *sepString* היא מחרוזת ריקה.

```
var x = String.removeAt
  (  "t=16, x=2, y=3, z=4",  0,  "," );       //   x = "x=2 ,y=3, z=4"
var x = String.removeAt
  (  "t=16, x=2, y=3, z=4",  5,  "," );     //   x = "t=16, x=2 ,y=3"
```

- (*string*, *elemString*, *number*, *sepString*) **replaceAt**. מחזירה מחרוזת חדשה המורכבת מ- *string* שבה אלמנט מספר *number* מוחלף ב- *elemString*. אלמנט הוא כל תת-מחרוזת של *string* המופרד על ידי התו הראשון של *sepString*. מפתח האלמנטים מתחיל מאפס.

אם *number* קטן מאפס, האלמנט הראשון מוחלף. אם *number* גדול ממספר האלמנטים שב- *string*, האלמנט האחרון הוא שמוחלף. אם *string* היא מחרוזת ריקה, מחרוזת חדשה המוגדרת כ- *elemString* מוחזרת. אם *number* הוא מספר נקודה צפה, תחילה הוא מומר לשלם בעזרת **Float.int** אם היא זמינה; אם לא, מוחזר **invalid**. **replaceAt** מחזירה **invalid** אם *sepString* היא מחרוזת ריקה.

```
var x = String.replaceAt
  (  "t=16, x=2, y=3, z=4", "v=12",  0,  "," );     //   x = "v=12, x=2 ,y=3, z=4"
var x = String.replaceAt
  (  "t=16, x=2, y=3, z=4", "v=11",  5,  "," );   //   x = "t=16,  x=2 ,y=3, v=11"
```

פונקציות המרה

ספריית String היא בעלת שתי פונקציות המרה:

- **format** (*fmtString*, *value*). ממירה ערך למחרוזת תוך שימוש במחרוזת התבנית *fmtString*. *fmtString* היא מחרוזת arbitrary שאמורה להכיל מציין תבנית טופס, אחד לפחות.

% [*width*] [. *precision*] *type*

type הוא "d" עבור שלם, "f" עבור נקודה צפה ו- "s" עבור מחרוזת. כדי להכליל סימן אחוז % בתבנית מחרוזת יש להשתמש בשני סימני אחוז רציפים ("%%").

width מציין את המספר המינימלי של תווים להדפסה. אם תוצאת התבנית קטנה מתווי *width*, ה-*value* המומר מלווה בתווים ריקים לשמאלו עד ש-*value* מגיע לרוחב *width*. אם מספר התווים ב- *value* גדול מ- *width*, או ש- *width* אינו מוגדר – כל התווים יודפסו.

precision מציין את דיוק התוצאה. הפירוש שלו תלוי ב-*type* של מפרט התבנית:

d – המספר המינימלי של הספרות בתבנית (ברירת המחדל היא אחד). התוצאה היא אפס, הממולא משמאל אם דרוש. אם *precision* הוא אפס, התוצאה היא מחרוזת ריקה.

f – מספר הספרות לימין הנקודה העשרונית של התוצאה, מעוגל לערך הנכון. אם *precision* הוא אפס, או רווח ללא ערך *precision* המופיע במציין התבנית, לא תופיע נקודה עשרונית בתוצאה.

s – המספר המקסימלי של תווים להדפסה (ברירת המחדל היא כל התווים).

format מחזירה **invalid** אם היא נתקלת במציין תבנית לא חוקי ב- *fmtString*, או כאשר היא אינה יכולה להמיר את *value* לסוג שבמציין התבנית.

```
var  x = String.format ( "%5d",  45 );                  // x = "   45"
var  x = String.format ( "%5.4d",  -45 );             // x = "  -45"
var  x = String.format ( "%f",  3.14159 );          // x = "3.14159"
var  x = String.format ( "%8.3f",  3.14159 );   // x = "   3.141"
```

- (*value*) **toString**. ממירה את *value* למחרוזת ומחזירה את התוצאה. פונקציה זו זהה בפעולתה להמרת סוגי נתונים אוטומטית של WMLScript, פרט לכך שהערך **invalid** מומר למחרוזת "invalid".

הספריה URL

הספריה URL מכילה קבוצת פונקציות המשמשות לאימות ולפעולה על URLs יחסיים או מוחלטים. תחביר URL המוגדר בפירוט ב- RFC 2396, הוא:

[scheme://] [host] [:port] [/]path [;parameters] [?query]
[#fragment]

עבור הפונקציות המחזירות מחרוזת המכילה מרכיב של URL, תוחמים (delimiters) קדמיים ואחוריים אינם כלולים בתוצאה המוחזרת. היוצא מן הכלל הוא שם הנתיב, הוא כולל אלכסונים ימניים.

לדוגמה, URL זה:

http://www.worldfaq.com:80/scripts/addcity.wml;3;2?loc=houston&g
mt=7#usa

מכיל את המרכיבים המפורטים בטבלה 5.5.

טבלה 5.5

Scheme	http
host	www.worldfaq.com
port	80
path	/scripts/addcity.wml
parameters	3;2
query	loc=houston,gmt=7
fragment	usa

ניתן לסווג את פונקציות הספריה URL לשלושה תחומים: פונקציות לניהול URLs, פונקציות להרחבת מרכיבים יחידים של URLs ופונקציה לאחזור תוכן מ- URL.

ניהול URLs

הספריה URL מאפשרת לך לאמת, לאחזר וליצור URLs.

- (*string*) **escapeString**. מחזירה מחרוזת, שווה ערך ל-*string* שבה כל תו מיוחד המזוהה ב- RFC 2396 מומר למספר ההקסדצימלי שלו, סימן אחוז ואחריו הקוד ההקסדצימלי בן שתי הספרות שלו. אם המחרוזת מכילה תווים שאינם חלק

מקבוצת תווי US-ASCII, מוחזר **invalid**. טבלה 4.4 מציגה את רשימת תווי החילופין של URL מ- RFC 2396.

```
Var  x  = URL.escapeString  ( "?x=2,y=3,z=4" );
// x= "%3fx%3d2%2cy%3d3%2cz%3d4"
```

- **getBase**(). מחזירה מחרוזת המכילה את ה-URL המוחלט, ללא החלק של יחידת ההוספה הנוכחית של WMLScript.
- **getReferer**(). מחזירה מחרוזת המכילה את ה-URL היותר קטן, ביחס ל-URL הבסיסי של יחידת ההוספה הנוכחית ולמקור שקרא לה. מחרוזת ריקה מוחזרת אם לא קיים כל ייחוס (referer).

לדוגמה, אם חפיסה ב-www.worldfaq.com/calcs/timecalc.wml קוראת לפונקציית WMLScript ב-www.worldfaq.com/scripts/timescript.wmls המכילה:

```
var  x  =  getReferer ( );
```

x מוגדר ל- "calcs/timecalc.wml"

- **isValid** (*string*). מחזירה **true** אם *string* הוא URL יחסי או מוחלט.
- **resolve** (*baseString, embeddedString*). מחזירה מחרוזת המכילה URL מוחלט, המהווה תוצאת שילוב של *baseString* ו-*embeddedString* בהתאם לכללים המצוינים ב- RFC 2396. אם *embeddedString* מכילה URL מוחלט, היא מוחזרת ללא שינוי.

```
var  root ="http://www.worldfaq.com";
var  path ="calcs/timecalc.wml";
set var  x  =  resolve  (  root,  path  );
// x = "http://www.worldfaq.com/calcs/timecalc.wml"
```

- **unescapeString** (*string*). מחזירה מחרוזת שוות ערך ל- *string*, ובה התווים המיוחדים המצוינים ב- RFC 2396 מומרים מהמספר ההקסדצימלי ל-US-ASCII שווי הערך שלהם. אם *string* מכילה תווים שאינם חלק מקבוצת התווים US-ASCII, מוחזר **invaild**. טבלה 4.4 מציגה את רשימת תווי החילופין של URL מ- RFC 2396.

```
var x = URL.unescapeString  ( "%3fx%3d2%2cy%3d3%2cz%3d4" );
// x = "?x=2,y=3,z=4"
```

פונקציות להרחבת מרכיבים

הפונקציות להרחבת מרכיבים מאפשרות לך לאחזר מרכיבים יחידים מ-URLs יחסיים ומוחלטים. פונקציות אלו בודקות תחילה האם ה-URL תקף, תוך שימוש ב-( )**isValid**, לפני הרחבת מרכיב כלשהו.

אם **isValid** היא **false**, הפונקציות הבאות מחזירות **invalid** :

- (*string*) **getFragment**. מחזירה מחרוזת המכילה את הקטע החלקי מ- *string* ה- URL המוחלט או היחסי.
- (*string*) **getHost**. מחזירה מחרוזת המכילה את שם ה-Host מ- *string* ה-URL המוחלט או היחסי.
- (*string*) **getPort**. מחזירה מחרוזת המכילה את מספר יציאת השרת מ- *string* ה- URL המוחלט או היחסי.
- (*string*) **getParameters**. מחזירה מחרוזת המכילה את הפרמטרים מ- *string* ה- URL המוחלט או היחסי.
- (*string*) **getPath**. מחזירה מחרוזת המכילה את הנתיב מ- *string* ה-URL המוחלט או היחסי.
- (*string*) **getScheme**. מחזירה מחרוזת המכילה את סכמת פרוטוקול אינטרנט מ- *string* ה-URL המוחלט או היחסי.
- (*string*) **getQuery**. מחזירה מחרוזת המכילה את חלק השאילתה מ- *string* ה- URL המוחלט או היחסי.

פונקציות לאחזור תוכן

פונקציית אחזור התוכן שבספריית URL מאפשרת לך להקצות את התוכן של קובץ טקסט למשתנה WMLScript.

(*urlString*, *contentTypeString*) **loadString**. מחזירה מחרוזת המכילה את התוכן המצוין על ידי *urlString* ו- *contentTypeString*. *contentTypeString* יכולה להכיל רק סוג תוכן אחד בצורה של "טקסט/תת-סוג" ללא רווחים לבנים בראש או מאחור. "תת-סוג" יכול להיות כל תת-סוג תקף. אם הטעינה נכשלה או התוכן המוחזר הוא מסוג שגוי, **loadString** מחזירה שלם המציין את קוד השגיאה, בהתבסס על סכמת URL. נעשה שימוש בקודי שגיאה של HTTP אם הסכמה היא HTTP או WSP.

כדי להשיג את קוד המקור של גוף פונקציית WMLScript "teststrings" ביחידת ההשלמה "testlib", תוכל להשתמש בקוד שלהלן :

```
var  loc = "http://www.worldfaq.com/test/testlib#teststrings";
var  x   = URL.loadString ( loc,  "text/wmlscript" );
```

הספריה WMLBrowser

הספריה WMLBrowser מכילה פונקציות המקנות לך גישה למשתני WML בהקשר של הסוכן-משתמש הנוכחי. בנוסף תוכל לומר לסוכן-משתמש WML לבצע משימה מסוימת כאשר מפרש WMLScript מסיים את פעולתו הנוכחית. כמו כן תוכל לאחזר את ה-URL היחסי של כרטיס WML המופעל נוכחית.

כל הפונקציות שלהלן מחזירות **invalid** אם הסוכן-משתמש אינו תומך ב-WML, או אם לא ניתן להפעיל את מפרש WMLScript באמצעות סוכן-משתמש WML.

פונקציות משתנה

פונקציות המשתנה מאפשרות לך לקרוא ולכתוב משתנים בהקשרו של הסוכן-משתמש WML הנוכחי.

- **getVar** (*string*). מחזירה מחרוזת המכילה את ערך המשתנה *string* מסוכן-המשתמש WML הנוכחי. **getVar** מחזירה מחרוזת ריקה אם המשתנה אינו קיים, ו-**invalid** אם *string* אינה מכילה שם משתנה בעל תבנית נכונה. *string* יכול להיות שם מילולי או שם משתנה.

- **setVar** (*string*, *value*). מגדירה את המשתנה *string* שבהקשר הנוכחי ל- *value*, ומחזירה **true** אם הפעולה הצליחה; אם לא, היא מחזירה **false**. **setVar** מחזירה **invalid** אם *string* אינו מכיל שם משתנה בעל תבנית נכונה. *string* יכול להיות שם מילולי או שם משתנה.

פונקציות משימה

- **go** (*urlString*). כאשר השליטה עוברת חזרה ממפרש WMLScript לסוכן-משתמש WML, **go** מאותתת לדפדפן WML, שעליו לטעון את החפיסה שצוינה על ידי *urlString*, כחלק מרצף ביצוע רגיל. **go** מחזירה **invalid** אם *urlString* אינה מכילה URL תקין. שם חלקי נחשב לשגוי – עליך לספק שם URL מלא.

  WMLBrowser.go ו- **WMLBrowser.prev** (ראה בהמשך) מבטלות האחת את השנייה - אם מי מהן נקראת לפני החזרת השליטה לסוכן-משתמש. רק הדרישה האחרונה מכובדת. אם **go**() או **prev**() מגדירות את ה-URL הבא למחרוזת ריקה, כל הדרישות מבוטלות.

- **prev** (). כאשר השליטה עוברת חזרה ממפרש WMLScript לסוכן-משתמש WML, **prev** מאותתת לדפדפן WML, שעליו לבצע את משימת **WMLBrowser.prev** ומחזירה **invalid** אם *urlString* אינה מכילה URL תקף. **WMLBrowser.go** (ראה קודם) ו- **WMLBrowser.prev** מבטלות האחת את השנייה - אם מי מהן נקראת לפני החזרת השליטה לסוכן-משתמש. רק הדרישה האחרונה מכובדת. אם **go**() או **prev**() מגדירות את ה-URL הבא למחרוזת ריקה, כל הדרישות מבוטלות.

- ( ) **refresh**. מאותתת לסוכן-משתמש WML שעליו לעדכן את הקשרו ולרענן את תצוגת ההתקן. לפונקציה זו יש את אותה ההשפעה שיש למשימה refresh. **WMLBrowser.refresh** מחזירה מחרוזת ריקה.
- ( ) **newContext**. מנקה את הקשר סוכן-משתמש WML הנוכחי, ומחזירה מחרוזת ריקה. פונקציה זו מתפקדת בדיוק כמו התכונה newcontext של כרטיס WML - הורקת מחסנית ההיסטוריה, הסרת כל המשתנים מוגדרי-ההקשר והגדרת ההתקן מחדש למצב ידוע (well-known state).



הערה!

WMLBrowser.newContext עלולה להיות פונקציה מאוד הרסנית. השתמש בה בזהירות.

פונקציית שאילתה

( ) **getCurrentCard**. מחזירה מחרוזת המכילה את ה-URL הקטן ביותר האפשרי של הכרטיס הנוכחי, המופעל על ידי הסוכן-משתמש WML, יחסית לבסיסה של יחידת ההוספה הנוכחית. הפונקציה מחזירה **invalid** אם לא קיים כרטיס נוכחי. **getCurrentCard** מחזירה URL מוחלט אם החפיסה המכילה את הכרטיס הנוכחי היא בעלת בסיס -URL השונה מזה של יחידת ההוספה הנוכחית.

לדוגמה, אם חפיסה ב-www.worldfaq.com/calcs/timecalc.wml#card1 קוראת לפונקציית WMLScript ב-www.worldfaq.com/scripts/timescript.wmls,

```
var x = getCurrentCard ( );
```

מגדירה את x ל- "calcs/timecalc.wml#card1".

הספריה Dialogs

הספריה Dialogs מכילה שלוש פונקציות ממשק משתמש. כל השלוש מציגות על מסך ההתקן הודעה בתבנית הנקבעת על ידי הסוכן-משתמש, וממתינות לתגובת המשתמש.

- (*string, defaultString*) **prompt**. מציגה מחרוזת ומבקשת (prompts) מהמשתמש להזין קלט. *defaultString* משמשת כערך ברירת המחדל של הקלט. הפונקציה מחזירה את מחרוזת הקלט של המשתמש.

  הקוד הבא עשוי ליצור את הבקשה המוצגת בתרשים 5.1 :

```
var  ssno ="200.30.4000";
var  ssno = Dialogs.prompt ("Social Security Number:  ", ssno );
```

```
Social Security No:
|

OK                ALPHA
```

תרשים 5.1 בקשה

```
How about a dance?

OK             Cancel
```

תרשים 5.2 דיאלוג אישור

```
Be sure to look both
ways.

OK
```

תרשים 5.3 אזהרה

- (*string, okString, cancelString*) **confirm**. מציגה מחרוזת ושתי ברירות למשתמש: OK ו-Cancel, תוך שימוש ב- *okString* וב- *cancelString* כתוויות הברירה, ואז ממתינה לבחירת המשתמש באחת הברירות. **confirm** מחזירה **true** אם המשתמש בחר ב-OK, ו-**false** אם המשתמש בחר ב-Cancel.

הקוד הבא ייצור את דיאלוג האישור המוצג בתרשים 5.2.

```
var OK = "Sure";
var Cancel = "No way!";
var result = Dialogs.confirm ( "How about a dance?",  OK,  Cancel  );
```

- (*string*) **alert**. מציגה את string, ממתינה לאישור המשתמש ומחזירה מחרוזת ריקה.

הקוד הבא ייצור את האזהרה המוצגת בתרשים 5.3.

```
result = Dialogs.alert ("Be sure to look both ways." );
```

פרק 6

i18N (בינלאומיות ועברית)

בינלאומיות (הנקראת גם i18N על ידי World Wide Web Consortium) היא נושא חשוב לסטנדרט דוגמת WAP, שעוצב לפעולה ברחבי העולם. עבור יישומי מחשב, התאמה אזורית היא מונח המשמש לתיאור הפיכת התוכנה למתאימה לארצות רבות. התאמה אזורית היא התהליך של שינוי התוכנה, כך שתהיה ניתנת לשימוש על ידי אנשים המבינים שפה מסוימת. בינלאומיות (Internationalization), אותה אנו מתארים בפרק זה, משמעותה התאמת תוכנה אחת (או שרת-תוכן, במקרה שלנו) לשרת לאומים רבים בו-זמנית.

ארכיטקטורת WAP מעוצבת לצורך תמיכה בקבוצות תווים רבות ושפות שונות. לצורך השגת מטרה זו, כמו בנושאים אחרים, WAP שאל רבות מהאינטרנט. לדוגמה, בשימוש ברישום IANA עבור קבוצות תווים וקודים המשמשת גם בכותרות HTTP. כמו כן אומץ השימוש ב-Unicode – קבוצת תווים אוניברסלית ההופכת במהירות לסטנדרט Web. WMLScript משתמשת ברצף ההשוואה של Unicode לצורך פעולה על מחרוזות.

ארכיטקטורת WAP מניחה את ההנחות הבאות לגבי סביבה תואמת WAP:

- סוכני-משתמש הם בעלי שפה מועדפת (המוגדרת על ידי המשתמש או על ידי ההתקן), ויכולים לקבל תוכן במיגוון קבוצות תווים ידועות.
- שרתי-תוכן יכולים לנפק תוכן בקבוצת תווים אחת או יותר.
- סוכני-משתמש ושרתי-תוכן יכולים לנהל את הקידוד המשמש בכל העברה שהיא ביניהם.
- כל סוגי התוכן והנתונים משודרים תוך שימוש בשיטות התומכות בהכרזת שפה וקידוד תווים.

תוך שימת לב למטרות והנחות אלו, עלינו להבהיר מספר מונחים בסיסיים קודם שנכנס לפרטיו של WAP i18N.

קבוצות תווים

קבוצת תווים (נקראת גם קוד תווים) היא מיפוי המגדיר קשר אחד לאחד בין אוסף של שמות תווים או מציינים, וקבוצה של שלמים חיוביים היכולים (אך אינם חייבים) להיות רציפים. קבוצת תווים אינה מכילה כל מידע אודות אופן תצוגת תו יחיד על מסך מחשב. היא ייצוג פנים-מחשבי של קבוצת תווים.

במהלך השנים הוגדר מיגוון של קבוצות תווים על ידי ארגוני-תקן וספקים. US-ASCII, אחת מן הראשונות המשמשות באינטרנט, עדיין נמצאת בשימוש נרחב. US-ASCII היא קוד בן שמונה סיביות, אך רק שבע מהן בשימוש. הסיבית היותר חשובה מוגדרת תמיד לאפס.

כיום, קבוצות התווים היותר נפוצות הן ISO Latin codes, IS0-8859-1 עד ISO-8859-9. (למידע נוסף אודות קבוצות התווים ISO, בקר ב-www.iso.ch). קבוצות אלו הן סדרה של קודי שמונה סיביות, שבהן 128 התווים הראשונים תואמים לקבוצת התווים US-ASCII ו- 128 התווים האחרים משמשים להגדרות מיוחדות בקבוצות של שפות שונות. RFC 2616 מגדיר את ISO-8859-1 כברירת המחדל של קבוצות התווים עבור מסמכי HTML וסוכני-משתמש, כפי שהוגדר בסטנדרט האינטרנט. סוכני-המשתמש אינם חייבים לתמוך בקבוצת תווים מלבד ISO-8859-1.

Unicode, קבוצת התווים הרשמית של WAP מכילה מספיק קודים ייחודים (מעל 65,000) כדי לכלול את כל התווים של השפות הכתובות והמדוברות בעולם. היא הופכת במהירות לסטנדרט המקובל באינטרנט. למעשה, מאז RFC 2070, Unicode (הנקראת גם UCS, **U**niversal **C**haracter **S**et), משמשת כהתייחסות כללית לתיאור נושאי תווים באינטרנט. ב-Unicode, 128 התווים הראשונים זהים ל-US-ASCII ו- 256 הקודים הראשונים זהים לאלה שב-ISO-8859-1.

סטנדרט ה-Unicode האחרון הוא Unicode 2.1. במאמץ לספק קבוצת תווים אחידה, ארגוני Unicode ו-ISO איחדו את מאמציהם. Unicode, עבור כל השימושים המעשיים, זהה ל- ISO 10646-1:1993. מידע נוסף על הסטנדרטים של Unicode ניתן למצוא ב-www.unicode.org.

קיים גוף האחראי לניהול הרישום של קודי התווים. נכון להיום (מרץ 2001) כולל רישום IANA Character Set כמעט 200 קבוצות תווים מוכרות וכינוייהן הרבים. טבלה 6.1 מכילה רשימה של קבוצות תווים הנמצאות בשימוש נפוץ באינטרנט, וכמה מהשפות שהן יכולות לייצג. מצייני קבוצות אותיות הינם רגישים לגודל אות.

טבלה 6.1 מספר קבוצות תווים הנמצאות בשימוש נפוץ באינטרנט ושפותיהן

קוד	**שפה**		
iso-8859-1	אפריקנס	אלבנית	בסקית
windows-1252	דנית	הולנדית	אנגלית
	פינית	צרפתית	גרמנית
	איטלקית	נורווגית	פורטוגלית
	סקוטית	ספרדית	שוודית
iso-8859-2	קרואטית	קאזחית	הונגרית
	פולנית	רומנית	סלוווקית
	סלוונית		
iso-8859-5	בולגרית	בילורוסית	מקדונית
	סרבית	אוקראינית	
iso-8859-6	ערבית		
iso-8859-7	יוונית		
iso-8859-8	**עברית**		
iso-8859-9	טורקית		
windows-1254			
koi-8-r	רוסית		
iso-8859-5			
shift_jis	יפנית		
iso-2022-jp			
euc-jp			

קידודי העברה

קידוד העברה (transfer encoding) מגדיר כיצד כל מספר של קבוצת אותיות מיוצג בסיביות, בעת העברתם דרך הרשת. באופן תאורטי, הגיוני להשתמש בקוד התווים לצורך קידודי ההעברה. קודי התו המוקדמים עדיין משמשים למטרה זו: ISO-8859-1 משודר, בדרך כלל, כקודים של שמונה סיביות המוגדרות על ידי קבוצת התווים ISO-8859-1. לעומת זאת, קיימות מערכות רשת שאינן יכולות לשדר או לקלוט בקלות ערכי נתונים של שמונה סיביות. במקום זאת, ניתן להשתמש בקידוד שבע סיביות כגון UUENCODE – טכניקת קידוד מוגדרת MIME.

ארגון Unicode הגדיר שני קידודי העברה עבור Unicode 2.1 : UTF-8 ו-UTF-16 (קיים גם קידוד UTF-7 שהוגדר על ידי ISO עבור העברות של שבע סיביות). קידודי UTF-16 זהים לקוד התוויות של Unicode 2.1. כל קוד Unicode 16 סיביות מיוצג על ידי ערך זהה לזה של UTF-16.

לרוע המזל, קודי תווים אינם תמיד יעילים מבחינת נפח. לדוגמה, קודי 256 התווים הראשונים ב-Unicode המשמשים לייצוג האותיות הנפוצות באינטרנט. אין זה הגיוני להשתמש בשני בתים לייצוג קודים שיכולים להסתפק בבית אחד. היעילות של גישה זאת היא של 50% לערך – עבור כל שני בתים, יש בית אחד ריק (או null).

כדי לפתור בעיה זו, ארגון Unicode הגדיר את קידוד UTF-8, המשתמש בבית אחד עד חמישה בתים לייצוג כל קודי התווים של Unicode. כפי שיש להניח ששיערת, ערכי בית-אחד נשמרים עבור 128 התווים הראשונים של קוד תווי US-ASCII. הערכים הנותרים מנסים להשתמש במספר הבתים הקטן ביותר עבור התווים הפופולריים. מכיון ש-WAP מעדיף להכניס את המידע הגדול ביותר במספר בתים הקטן ביותר עבור תשדורת סלולרית, WAP Forum ממליץ ומעודד את השימוש ב-UTF-8 כקוד ברירת המחדל של העברה.

ניתן להשתמש ב-UTF-8 עבור כל השפות. זהו הקידוד המומלץ באינטרנט והתמיכה בשימוש בו הולכת וגדלה.



הערה!

אם תקרא את מסמכי הסטנדרטים השונים והדרישות להערות (Requests for Comments (RFCs)) המגדירים את קידוד קבוצות תווי האינטרנט וההעברות, תמצא שמכיון שלא קיים מינוח סטנדרטי, הם מבלבלים לא מעט. דוגמה אחת היא רישום התווים IANA. הוא כולל ערך עבור Unicode, שהוא קוד תווים, ועבור UTF-8 המהווה קידוד ספציפי של Unicode. היה זה ברור לו היו משתמשים ב-UTF-16 ככינוי ל-Unicode, או כשם חליפי, אולם לא כך נעשה. כמו כן RFC 2616 משתמש במונח "קבוצת תווים" (characterset) כדי לתאר קידוד העברה. דבר זה הוא ודאי מכיון שהתכונה Charset של HTTP, המתוארת בהמשך הפרק, משמשת לצורך הגדרת קידודים. למרבה המזל, תודות ל-Unicode, מספר קודי התווים שעליך לדעת אודותם, הולך ופוחת.

כמתכנתים, עלינו להכיר קבוצות תווים וקידוד. סוכני-משתמש WAP אמורים להבין Unicode, קוד התווים הרשמי של WAP, אך איזה קידוד? ברירת המחדל המקובלת היא UTF-8, משום יעילות צריכת הנפח שלו, אולם אין היא מחייבת. סוכני-משתמש WAP יכולים להשתמש גם בקידודים אחרים. בחירה בולטת היא ISO-8859-1, קבוצת התווים הלטינית הנפוצה. עלינו לדעת כיצד לחשב מהו הקידוד המועדף על ידי סוכן-משתמש, ולהשתמש בו אם רק ניתן.

בנוסף להתמודדות עם בעיות קבוצת התווים שבשימושו של הסוכן-משתמש, אין אנו יכולים לבחור את קבוצת התווים המקומית או את הקידוד עבור השרתים המקוריים שלנו. ההחלטה נעשית לרוב ברמת מערכת ההפעלה או שפת התכנות. לדוגמה, ה-Java servlets מקדדים מחרוזות ב-Unicode UTF-16. לעומת זאת, מספר שרתים משתמשים ב-US-ASCII או ב-ISO-8859-1. סוכני-משתמש WAP אינם נדרשים להבין קבוצות תווים אלו. אם אמנם כך הוא, כיצד נוכל לנהל את התקשורת כדי שהסוכן-משתמש יקבל את הקידוד הנכון?

למרבה המזל, כפי שתראה בהמשך הפרק, קיימות דרכים לטפל בבעיות אלו. ברמה אחת, אתה כמתכנת יכול לקבוע את סוג הקידוד שיועדף על ידי הסוכן-משתמש, ולהעביר את הקידוד הזה על ידי יצירת תוכן בקידוד הנכון. פתרון פשוט יותר הוא להניח ל- WAP gateways לפתור את הבעיה. רוב WAP gateways יכולים לנטר את התקשורת בין הסוכן-משתמש לשרת-התוכן, תוך פענוח סוג הקידוד שכל צד מבין, ולבצע את קידוד ההעברה עבורך. הדבר משחרר אותך מהצורך לטפל בבעיה, רק ודא שאתה מודע ליכולות השרת שלך.

שפות

כפי שעליך להיות מודע לקבוצת התווים ובעיות הקידוד כדי לאפשר למערכת WAP שלך לפעול היטב, עליך לדאוג למשתמשי הקצה. עליך לוודא שמשתמשי הקצה יוכלו לקרוא ולהבין את התוכן הנשלח אליהם.

שפה, אין לה כל קשר לקבוצות תווים או מחשבים, היא דיאלקט מדובר. לדוגמה, רק במקרה, ISO-8859-5 משמשת לקידוד רוסית. לעומת זאת, מזהה שפה יכול להיות מועיל לצורך זיהוי תכונות מסוימות של השפה, כאשר מחליפים תוכן בתצוגת מחשב.

דוגמה בולטת היא כיוון סידור התצוגה. שפות מערביות מוצגות ונקראות משמאל לימין. לעומת זאת, יפנית מסודרת ונקראת מימין לשמאל וכמוה גם עברית. כמו כן, ייתכן שיהיו לשפה כללים מסוימים המגדירים כיצד לשלב קודי תווים רבים לישויות הנראות על המסך. תכונות אלו הן אפיוני השפה, ולא אפיוני קבוצת התווים או הקידוד.

HTTP מגדיר קבוצה של תגיות שפה הניתנות לשימוש על ידי סוכני-משתמש ושרתי-תוכן, כדי להתאים את השפה הרצויה לצורך ההעברה. תגיות של HTTP מורכבות מרשימה של קיצורי השפה העיקריים (המורכבים משני תווים מתקן ISO639) ומקיצור דיאלקט אופציונלי (בן שני תווים מתקן ISO3166). תגית שפה טיפוסית יכולה להיראות כך:

it, it-va. fr-fx, fr, en, iw

תגית זו מזהה את השפות הבאות לפי סדר חשיבותן: איטלקית, איטלקית של הוותיקן, צרפתית עירונית, צרפתית, אנגלית ועברית (iw). תוכל למצוא פרטים נוספים על תגיות שפה של HTTP ב- RFC 1766.

קיים מרכיב נוסף אחד של i18N שחייבים להזכירו: תצוגה (rendering) – הדרך בה קבוצת תווים נראית על תצוגת ההתקן. התצוגה היא בלתי תלויה בקודי תווים, בקידוד העברית ובשפות (זו הסיבה לכך שהגדרת קבוצת תווים היא מיפוי בין שמות תו ושלמים ייחודיים, לא תרגומי תווים). היא כוללת גופנים, הגדרות סגנון (כגון נטוי ומודגש) ויכולות התצוגה של ההתקן.

ביישומים מבוססי Web טיפוסיים, על משתמשי קצה לוודא שהגדרת המערכת שלהם מתאימה להצגת קבוצות תווים ספציפיות. לדוגמה, אין טעם בהגדרת דפדפן ה-Web שלך, כך שיוכל לדרוש ולקבל מסמכים יפניים, אם אין גופנים יפניים במערכת שלך.

ביישומים מבוססי WAP, אין זו בדרך כלל בעיה קריטית, רק שעליך להיות מודע לה. בהתקני WAP, קבוצת התווים וטכנולוגיית התרגום קשורים יחד. אם סוכן-משתמש WAP מאפשר לך להגדיר קבוצת תווים יפנית כברירת מחדל (או שהוא הגדיר זאת עבורך, אוטומטית), תוכל להיות בטוח שהוא יכול להציג תווים יפניים.

שרתי-תוכן רב-לשוניים

אם ברצונך להפעיל אתר רב-לשוני, תוכל לבצע זאת בקלות רבה ב-WAP, תוך שימוש בידיעותיך על כותרות HTTP. ישנן שתי כותרות חשובות: **Accept-Language** ו-**Content-Language**. **Accept-Language** היא כותרת דרישה המוגדרת על ידי סוכן-משתמש WAP באחת משתי דרכים: הגדרה אוטומטית, במקרה שהוא מכיר רק שפה אחת, או שהסוכן-משתמש מגדיר את הכותרת כתגובה על ציון השפה המועדפת על ידי המשתמשים. כותרת **Accept-Language** טיפוסית נראית כך:

```
Accept-Language: it,  it-va,  fr, en
```

מציינת שהמשתמש (או אולי הסוכן-משתמש) מעדיף איטלקית ואחריה איטלקית של הוותיקן, אחריה צרפתית ולבסוף אנגלית. זוהי משימתו של שרת-התוכן שלך לקבל את תגית השפה הזו, לפענח אותה ולהחזיר את התוכן המתאים.

אם מדובר בתוכן סטטי, השיטה בה תשתמש כדי לאחזר את ערך הכותרת **Accept-Language** תלויה בשרת שלך. לדוגמה, בשרת Apache Web הפופולרי, ניתן להגדיר את מבנה התיקיה כך שמשתמשים מופנים אוטומטית למסמכים הכתובים בשפה הנכונה, בהתבסס על ערך הכותרת **Accept-Language**. הפרטים כיצד לבצע זאת הם מעבר לתחומו של ספר זה. בדוק את תיעוד שרת ה-Web שלך.

במקרה של תוכן דינמי, עליך להשיג את הערך של **Accept-Language**. בתוכניות CGI, משתנה הסביבה **HTTP_ACCEPT_LANGUAGE** מוגדר אוטומטית לערך הכותרת **Accept-Language**. עבור Java servlets, תוכל לדרוש את הגדרת הכותרת תוך שימוש במשפט דוגמת זה שלהלן:

```
String language=req.getHeader ( "Accept-Language" );
```

משאתה יודע את העדפת המשתמש (או הסוכן-משתמש), עליך ליצור את התוכן, ולאותת לסוכן-משתמש המקבל את התגובה, מהי השפה המשמשת לתגובה. יש שתי דרכים לעשות זאת.

הדרך הראשונה, המתאימה לחפיסות סטטיות ולתוכניות CGI, בהן אינך יכול להגדיר ערכי כותרת כלשהם, היא להשתמש באלמנט <**meta**>WML. ודאי זכור לך שאלמנט <**meta**>, הניתן לשימוש רק במסגרת אלמנט החפיסה <**head**>, מאפשר לך להגדיר תגיות **http-equiv**. תגית **http-equiv** מורה ל- WAP gateway שעליו ליצור כותרת HTTP עבורך, ולהוסיף אותה לתגובה.

לדוגמה, אם אתה בונה חפיסת WML סטטית או דינמית בצורה הבאה:

```
<wml>
   <head>
      <meta http-equiv="Content-Language"  content="EN"/>
   </head>
    '
    ' rest of the deck
    '
</wml>
```

ה-Gateway מפריד את אלמנט <**meta**> מהחפיסה לפני שהוא מהדר אותה. ככלות הכל, מדוע לשלוח מידע חיצוני חלקי דרך קשר סלולרי בעל רוחב-פס צר? במקום זאת, הוא מוסיף את הכותרת הבאה בחזית החפיסה:

```
Content-Language: EN
```

הכותרת מודיעה לסוכן-משתמש מהי השפה של מסמך התגובה, כך שיוכל לטפל בה כיאות.

אם תוכנת יצירת התוכן הדינמי שלך מאפשרת לך גישה לכותרות התגובה, כפי ש-Java servlets מאפשרים, תוכל לדלג על אלמנט <**meta**>, וליצור במקומו את כותרת **Content-Language** הנכונה. שיטה זו עדיפה על השימוש באלמנט <**meta**> – היא מהירה יותר, היא צורכת פחות רוחב-פס ולא כל ה-Gateways מכירים את אלמנט <**meta**>.

להלן דוגמה פשוטה של Java servlet "Hello World", הלוקח את הכותרת **Accept Language** מהדרישה, מוצא מהי השפה בה עליו להשתמש ומחזיר את התגובה הבינלאומית המתאימה, IntHello.java.



הערה!

כדי שקבצי ה-Java יעבדו יש להדר אותם. קרא על כך בקובץ ReadMe1st.txt בתקליטור המצורף.

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.util.*;

public class IntlHello extends HttpServlet {

  static String EOL = "\r\n";

  public void doGet ( HttpServletRequest req, HttpServletResponse res)
    throws ServletException, IOException
  {

    res.setHeader      ( "Content-location", "http://www.worldfaq.com/servlet/intlhello" );

 // Figure out what language we have, default is English.

    String language = req.getHeader ( "Accept-Language" );
    String message  = "Hello World";
    String charset  = "ISO-8859-1";

    if ( language != null ) {
// If there is a language header with a value we can handle,
// change the message to reflect the proper dialect.

      language = language.toUpperCase ();

      if ( language.indexOf ( "FR" ) != -1 ) {
        message = "Bonjour monde!";
      }
      if ( language.indexOf ( "ES" ) != -1 ) {
        message = "Hola mundo!";
      }
       if ( language.indexOf ( "DE" ) != -1 ) {
        message = "Hallo welt!";
      }
    } else language = "EN";

// Set the language response header and content type,
// including the charset attribute.

    res.setHeader      ( "Content-language", language );
    res.setContentType ( "text/vnd.wap.wml; charset=" + charset );
```

```
// Output the Hello World deck.

    ServletOutputStream out = res.getOutputStream();
    out.println (
      "<?xml version=\"1.0\"?>" +
      "<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML 1.1//EN\"" + EOL +
      "\"http://www.wapforum.org/DTD/wml_1.1.xml\">" + EOL +
      "<wml>" + EOL +
      "<head>" + EOL +
      "<meta http-equiv=\"Content-Language\" content=\""
                                                   + language + "\"" + "/>" + EOL +
      "</head>" + EOL +
      "<card id=\"multilingual\" >" + EOL +
      "<p>" + EOL +
      message + EOL +
      "</p>" + EOL +
      "</card>" + EOL +
      "</wml>"
    );

}   }
```

שים לב שלא לקחנו שום סיכונים. השתמשנו באלמנט **<meta>**, וגם הגדרנו כותרת תגובה **Content-Language** כדי להבטיח שהסוכן-משתמש ידע מה שלחנו.

כאשר תפעיל את ה-servlet הזה, תוך שימוש ב-Telnet ובכותרת **Accept-Language** המוגדרת ל-"fr":

```
GET /servlet/intlhello  HTTP/1.1
accept-language:  fr
```

תקבל את התגובה הבאה (הוספנו את ההזחה לצורך בהירות):

```
HTTP/1.1 200 OK
Server:Zeus/3.1
Date:Thu, 03 Jan 1999 23:48:09 GMT
Connection: close
Content-Location: http://www.worldfaq.com/servlet/intlhello
Content-Type: text/vnd.wap.wml; charset=iso-8859-1
Content-Language: FR

<?xml version="1.0"?><!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
    "http://www.wapforum.org/DTD/wml_1_1.xml">
<wml>
    <head>
        <meta http-equiv="Content-Language" Content="FR" />
    </head>
```

```
<card id="multilingual">
    <p>
        Bonjour monde!
    </p>
</card>
</wml>
```



הערה!

הכותרת Content-Type מכילה פרמטר נוסף, charset, שיתואר בהמשך.

ביכולתך להגדיר עדיפויות שפה עבור אלמנטים נפרדים של חפיסת WML. הדבר יקנה לך יתר גמישות אם תשלח תוכן רב-לשוני.

התכונה **xml:lang** היא תכונה הניתנת לשימוש עם אותם אלמנטי WML המכילים תוכן. התכונה **xml:lang** מורה לסוכן-משתמש לעשות כמיטב יכולתו לתרגם כראוי את התוכן המתויג בתכונה **xml:lang**.

מכיון של-WML יש אלמנטים מקננים, תוכל גם לקנן תכונות **xml:lang**.

מפרט WML 1.1 ממליץ שסוכני-משתמש ישתמשו בקדימויות שלהלן (לפי הסדר) בעת זיהוי שפתו של אלמנט:

- התכונה **xml:lang** של האלמנט.
- תגית **xml:lang** של ההורה הקרוב ביותר.
- כל מידע שפה הכלול בכותרות התגובה (כותרת **Content-Language**) ו-meta data.
- עדיפויות ברירת המחדל של המשתמש.

חפיסת התגובה הקודמת יכולה להיכתב מחדש כדלהלן:

```
<wml>
  <card  id="multilingual"  xml:lang="FR" >
    <p>
       Bonjour monde!
    </p>
  </card>
</wml>
```

מאחר שלתכונה **xml:lang** של הכרטיס יש קדימות על פני כל תכונות התגובה, נוכל להסיר את התגית <**meta**> ואת הכותרת **Content-Language**.

Charset ותגובות

בעת טיפולך בהתאמת שפה בין הסוכן-משתמש ושרת-התוכן, עליך לשים לב לקבוצות התווים המעורבות. כדוגמת התאמת שפה בין סוכן-משתמש ושרת-תוכן, הפעולה היא לשני הכיוונים. ראשית, בעת שליחת מסמכים עליך לוודא שאתה שולח קבוצת תווים בת-שימוש משרת-התוכן לסוכן-משתמש. חשוב באותה מידה: כאשר הסוכן-משתמש שולח אליך תוכן, עליך לוודא שאתה מבין אותו. אתה שולט בראשון ממנגנונים אלה בעזרת התכונה **Charset** של הכותרת **Content-Type**. ואילו בשני – על ידי הבנת התכונות **accept-charset**, הנשלחות אליך על ידי הסוכן-משתמש.

בעת שליחת תוכן לסוכן-משתמש, עליך לוודא שהסוכן-משתמש יכול להבין ולטפל כיאות בקבוצת התווים שבכוונתך לשלוח. לרוב, כל סוכני-המשתמש WAP מבינים Unicode, ומצפים לקבל אותו בקידוד UTF-8. אולם עליהם להכריז באופן ייחודי אודות יכולת זו, באמצעות כותרת דרישה **Accept-Charset** הנראית כדוגמה הבאה:

Accept-charset: UTF-8, utf-8, *

הדבר מורה לשרת-התוכן שהסוכן-משתמש מסוגל לקבל תגובות המקודדות ב-UTF-8. משמעות הכוכבית היחידה היא שהסוכן-משתמש יקבל כברירה אחרונה כל קידוד שהוא. ייתכן שהוא ייכשל ויציג הודעת שגיאה, אך קודם לכן הוא ינסה לטפל בהודעה בדרך הרגילה.

משאתה יודע שהסוכן-משתמש יכול לטפל בקידוד UTF-8, עליך להיות מסוגל לשלוח חזרה כל קידוד מקובל (mainstream encoding). הסוד מאחורי הפעולה הוא ש-WAP gateways ממיר את קוד התוכן שלך ל-UTF-8 לפני שליחתו לסוכן-משתמש. עליך רק לוודא שה-Gateway יכול לנהל את כל המרות הקוד שאתה עשוי להזדקק להן.

אם הסוכן-משתמש אינו מטפל בקידוד UTF-8, עליו לשלוח אליך כותרת קבוצת תווים יותר ספציפית, המציינת את קבוצת התווים הרצויה. לדוגמה, אם הדרישה כוללת את הכותרת הבאה:

Accept-charset: ISO-8859-7

המשמעות היא שהסוכן-משתמש עוצב עבור השוק היווני, ועדיף שתגיב בקידוד ISO-8859-7 כדי שהתוכן שלך יוצג כראוי על מסך ההתקן.

כאשר אתה יודע שהסוכן-משתמש מעדיף קידוד ISO-8859-7, ביכולתך ליצור את התוכן היווני בעצמך, תוך שימוש בקודי ISO-8859-7 במסמכי המקור ותוכניות CGI שלך. כמו כן, ביכולתך להטיל על ה-Gateway לבצע את העבודה עבורך. לדוגמה, באמצעות יצירת תוכן ב-Unicode(UTF-16) תוך שימוש ב-Java servlet, ובהוספת כותרת **Content-Type** לחפיסת התגובה, כמו בדוגמה שלהלן:

Content-type: text/vnd.wap.wml; charset=ISO-8859-7

המסמנת (flagging) את התוכן כתוכן ISO-8859-7 (עד כה, לא הצגנו צורה זאת של כותרת **Content-Type** המכילה את התכונה האופציונלית **charset**).

ה-Gateway אמור להמיר את Unicode לקוד התווים ISO-8859-7 לפני שליחתו חזרה אל הסוכן-משתמש. בדוק את מפרט ה-Gateway כדי לוודא את תמיכתו בהמרת הקוד הדרושה לך.

כפי שכבר דנו, הדרך להוספת כותרת **Content-Type** לתגובה שלך תלויה בשרת-התוכן שלך. בחפיסות סטטיות, אתה עשוי להגדיר את תצורת השרת שלך, כך שיוסיף אוטומטית את כותרת **Content-Type** הנכונה בהתבסס על שם המסמך או מיקום התיקיה שלו. עבור חפיסות סטטיות ודינמיות גם יחד, תוכל להשתמש בתגית <**meta**> בכותרת החפיסה. לדוגמה, האלמנט <**meta**> :

```
<deck>
  <head>
   <meta http-equiv="Content-type"
      content="text/vnd.wap.wml;  charset= ISO-8859-7 "
   </>
  </head>
```

מוסר ל- WAP gateway מידע זהה לכותרת **Content-Type** הקודמת. ה-Gateway אמור להכיר את האלמנט <**meta**> ולבצע כל המרת קידוד דרושה.

לסיום, עבור מנגנוני CGI כגון Java servlets, ביכולתך ליצור ולשנות את כותרות התגובה ב-servlets שלך. כפי שציינו קודם, שיטה זו עדיפה על פני השימוש באלמנט <**meta**>.



הערה!

כאשר מדברים על מסירת תוכן וקידוד, חשוב להבין שהקידוד אינו נוגע להודעת התגובה כולה, אלא רק לגוף ההודעה. כותרות התגובה כולן נשלחות ב-US-ASCII כפי שמצוין במפרט HTTP 1.1. פירוש הדבר הוא, שאם תגיב לדרישה באמצעות הודעת MIME רבת חלקים, תוכל לקודד בנפרד כל חלק של ההודעה בכותרות Content-Type.

כדי להדגים כמה מהרעיונות שבפרק זה, יצרנו דוגמת יישום, הנקראת **ShowCharSet**, והכוללת חפיסת WML סטטית ו-Java servlet. החפיסה מאפשרת תצוגה של מספר דוגמאות בכל אחת מקבוצות התווים של ISO-Latin, לפי דרישת המשתמש. התגובה היא בצורת טבלה קטנה ובה מספר תווים ב- 128 קודי התווים העליונים של כל קבוצת תווים, כדי להדגים את השוני שביניהם.

תרשים 6.1 מציג את המסך הראשי עבור ה-ShowCharSet.wml. תרשים 6.2 מציג את אשר קורה אם אתה בוחר בקבוצת התווים ISO-8859-7.

להלן חפיסת WML עבור **ShowCharSet**:

```
<wml>
  <card id="charsetselect" >
    <p>
     Select a char set:
     <select>
       <option onpick="../servlet/showcharset?charset=ISO-8859-1">ISO-8859-1
</option>
       <option onpick="../servlet/showcharset?charset=ISO-8859-2">ISO-8859-2
</option>
       <option onpick="../servlet/showcharset?charset=ISO-8859-3">ISO-8859-3
</option>
       <option onpick="../servlet/showcharset?charset=ISO-8859-4">ISO-8859-4
</option>
       <option onpick="../servlet/showcharset?charset=ISO-8859-5">ISO-8859-5
</option>
       <option onpick="../servlet/showcharset?charset=ISO-8859-6">ISO-8859-6
</option>
       <option onpick="../servlet/showcharset?charset=ISO-8859-7">ISO-8859-7
</option>
       <option onpick="../servlet/showcharset?charset=ISO-8859-8">ISO-8859-8
</option>
       <option onpick="../servlet/showcharset?charset=ISO-8859-9">ISO-8859-9
</option>
     </select>
    </p>
  </card>
</wml>
```

```
Select a char set:
1 ▶ ISO-8859-1
2   ISO-8859-2
3   ISO-8859-3
4   ISO-8859-4
5   ISO-8859-5
OK
```

תרשים 6.1 המסך הראשי של ShowCharSet

```
ISO-8859-5
&#xb0  А    &#xb1  Б
&#xc0  Р    &#xc1  С
&#xd0  а    &#xd1  б
&#xe0  р    &#xe1  с
&#xf0  №    &#xf1  ё
OK
```

תרשים 6.2 דוגמת קבוצת התווים ISO-8859-5

```
ISO-8859-7
&#xb0  °    &#xb1  ±
&#xc0  ΐ    &#xc1  Α
&#xd0  Π    &#xd1  Ρ
&#xe0  ΰ    &#xe1  α
&#xf0  π    &#xf1  ρ

OK
```

תרשים 6.3 דוגמת קבוצת התווים ISO-8859-7

זהו אלמנט <**select**> יחיד, המכיל תשעה פריטים: פריט אחד לכל אחת מקבוצת התווים שב-ISO Latin character sets. כאשר המשתמש בוחר באחת האפשרויות, נקרא ה-servlet **ShowCharSet**, ושם קבוצת התווים מועבר בפרמטר **CharSet**. ה-servlet מקבל את ערך הפרמטר, ושולח חזרה את התגובה המכילה את תצוגת ההדגמה.

למעשה, ה-servlet מבצע יותר מכך. ראשית, הוא בודק האם הסוכן-משתמש תומך בקידוד UTF-8. אם לא, הוא מחזיר הודעת שגיאה במקום דוגמה של קבוצת התווים. כמו כן, הוא מוודא האם הפרמטר **charset** קיים – בדיקה שתחזיר אמת, אם חלק ה-WML של היישום מקודד נכון. אם לא קיים פרמטר **charset**, תוחזר הודעת שגיאה. אם ה-servlet עובר בהצלחה את שתי בדיקות השגיאה, הוא שולח חזרה חפיסה עם כותרת **Content-Type** הנכונה, כולל התכונה **charset** שנשלחה אליו.

להלן קוד המקור של ה-Java servlet, הקובץ ShowCharSet.java:

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.util.*;

public class ShowCharSet extends HttpServlet {

  static String EOL = "\r\n";

  public void doGet (

    HttpServletRequest req,
    HttpServletResponse res)
    throws
    ServletException,
    IOException
  {
```

```
// Make sure the user agent understands UTF-8. That means it can
// handle all character sets. Look for the Accept-Charset header.
// With a UTF-8 setting.

    String charSets = req.getHeader ( "Accept-Charset" );
    boolean foundUTF8 = false;

    if ( charSets != null ) {
      StringTokenizer parser = new StringTokenizer ( charSets, "," );
      String token;
      while ( parser.hasMoreTokens () && ( !foundUTF8 )) {
        token = parser.nextToken ();
        token = token.toUpperCase ();
        if ( token.indexOf ( "UTF-8" ) != -1 ) foundUTF8 = true;
    } }

// Get the user-requested character set from the GET charset
// variable and output the response headers.

    String respCharSet = req.getParameter ( "charset" );
    if ( respCharSet == null )
      res.setContentType ( "text/vnd.wap.wml" );
    else
      res.setContentType ( "text/vnd.wap.wml; charset=" + respCharSet );

    res.setHeader ( "Cache-Control", "must-revalidate" );
    res.setHeader ( "Content-location",
            "http://www.worldfaq.com/servlet/showcharset" );

 // Output the beginning of the resposne message deck.

    ServletOutputStream out = res.getOutputStream();
    out.println (

      "<?xml version=\"1.0\"?>" +
      "<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML 1.1//EN\"" + EOL +
      "\"http://www.wapforum.org/DTD/wml_1.1.xml\">" + EOL +
      "<wml>" + EOL +
      "<card id=\"showcharset\" >" + EOL +
      "<p>"
    );

// If there's no Accept-Charset=UTF-8 request header, error.
```

```
    if ( ! foundUTF8 ) {
      out.println (
        "This device does not accept UTF-8 encoding. "
      );
    }

// If there's not user-requested character set parameter
// (which shouldn't happen), error.

    else if ( respCharSet == null ) {
      out.println (
        "Sorry, no response character set was specified."
      );
    }

// No errors, return a sample of the requested character set.
// These codes will be transcoded by the gateway into UTF-8.

    else {
      out.println (

        respCharSet + "<br/>" + EOL +
        "<table columns=\"4\">" + EOL +

        "<tr>" + EOL +
          "<td>&amp;#xb0 </td>" + EOL +
          "<td>&#xb0; </td>" + EOL +
          "<td>&amp;#xb1 </td>" + EOL +
          "<td>&#xb1; </td>" + EOL +
        "</tr>" + EOL +

        "<tr>" + EOL +
          "<td>&amp;#xc0 </td>" + EOL +
          "<td>&#xc0; </td>" + EOL +
          "<td>&amp;#xc1 </td>" + EOL +
          "<td>&#xc1; </td>" + EOL +
        "</tr>" + EOL +

        "<tr>" + EOL +
          "<td>&amp;#xd0 </td>" + EOL +
          "<td>&#xd0; </td>" + EOL +
          "<td>&amp;#xd1 </td>" + EOL +
          "<td>&#xd1; </td>" + EOL +
        "</tr>" + EOL +
```

```
      "<tr>" + EOL +
        "<td>&amp;#xe0 </td>" + EOL +
        "<td>&#xe0; </td>" + EOL +
        "<td>&amp;#xe1 </td>" + EOL +
        "<td>&#xe1; </td>" + EOL +
      "</tr>" + EOL +

      "<tr>" + EOL +
        "<td>&amp;#xf0 </td>" + EOL +
        "<td>&#xf0; </td>" + EOL +
        "<td>&amp;#xf1 </td>" + EOL +
        "<td>&#xf1; </td>" + EOL +
      "</tr>" + EOL +
      "</table>"
    );
  }

// Finish up the response deck.

    out.println (
      "</p>" + EOL +
      "</card>" + EOL +
      "</wml>"
    );
}  }
```

Charset ודרישות

בדרך כלל, הגדרות **charset** משמשות את הסוכן-משתמש לצורך הכללת קבוצת התווים המועדפת עליו בתגובה המתקבלת משרת-התוכן. תופעת לוואי מעניינת של כותרת הדרישה **Accept-Charset** היא שכל נתוני **POST** הנכללים בדרישה, מקודדים לקוד קבוצת התווים שב- **Accept-Charset** לפני שליחתם אל שרת-התוכן.



הערה!

נתוני GET המקודדים יחד עם דרישת HTTP, אינם מקודדים מחדש בהתאם לכותרת Accept-Charset לפני שהם נשלחים.

לעומת זאת, עשויים להיות מקרים בהם לא תרצה שנתוני **POST** יומרו לקבוצת התווים המועדפת על ידי הסוכן-משתמש. במקום זאת, תרצה לציין קבוצת תווים אחרת.

למרבה המזל, אלמנט <**go**> WML הוא בעל התכונה האופציונלית **Accept-Charset**, המאפשרת לך להגדיר מהי קבוצת התווים שנתוני **POST** יקודדו בה. תכונה זו נשלחת ל- WAP gateway שכבר מבצע את הקידוד, ומעביר את הנתונים המקודדים אל שרת-התוכן כנתוני **POST**. מכאן ואילך זוהי סמכותו של שרת-התוכן להחליט מה לעשות.



הערה!

*לתכונה Accept-Charset של אלמנט <**go**> WML, עדיפות על פני כל ערך שהסוכן-משתמש עשוי להקצות לכותרת Accept-Charset שבדרישה.*

חשוב ששרת-התוכן ידע שהוא מקבל נתונים בקבוצת תווים לא צפויה. כדי להזהירו, WAP gateway מוסיף את הגדרת קבוצת התווים החדשה לכותרת **Content-Type** של הדרישה היוצאת. הכותרת תיראה בערך כך:

```
Content-type: application/x-www-forum-urlencoded; charset=ISO-8859-5
```

כמו כן, באפשרותו של ה-gateway לשנות את כותרת-הדרישה **Accept-Charset**, כדי לשקף את קבוצת התווים של הנתונים.

להלן יישום WML/Java servlet פשוט הנקרא **TransMsg**, המאפשר לך לבחור אחת משלוש קבוצות תווים, להזין הודעה, ואז לשלוח את ההודעה במשתנה **POST** בעל התכונה **accept-charset** המוגדרת לקבוצת התווים שבחרת. הפונקציה אינה מעניינת במיוחד, אולם כתוצאה מפעולתה, ניתנת לך האפשרות להביט בכותרות הדרישה ופרמטרים, כדי לראות את תוצאת התכונה **accept-charset**.

להלן חלק ה-WML של היישום TransMsg.wml:

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">

<wml>

  <head>
    <meta http-equiv="cache-control" content="no-cache" />
  </head>

<!-- select a character set -->

  <card id="charset" >
    <do type="accept" >
      <go href="#message" />
    </do>
    <p>
```

```
    Select a char set:
    <select name="charset" value="ISO-8859-1" >
      <option value="ISO-8859-1">ISO-8859-1 </option>
      <option value="ISO-8859-7">ISO-8859-5 </option>
      <option value="ISO-8859-7">ISO-8859-7 </option>
    </select>

  </p>
 </card>

<!-- get the message and translate it-->

 <card id="message" >
  <do type="accept" >
    <go href="http://www.worldfaq.com/servlet/transMsg"
       method="post" accept-charset="$(charset)" >
      <postfield name="message" value="$(message)" />
<!--    <postfield name="charset" value="$(charset)" /> -->

    </go>
  </do>
  <p>
  Enter a message:
  <input name="message" />

  </p>
 </card>

</wml>
```

שים לב, שבנוסף לשליחת ההודעה, אנו שולחים את בחירת קבוצת התווים במשתנה **charset**. אנו בודקים משתנה זה, ושולחים את ההודעה חזרה אל הסוכן-משתמש, תוך שימוש בקבוצת התווים הנדרשת. כך מתאפשר לנו לראות את תוצאות ההודעה המקודדת. אנו משתמשים במשתנה **charset** רק כדי להקל על עבודתנו. הדבר חוסך לנו את הטרחה לכתוב קוד המפענח את הכותרות **Content-Type** ו-**Accept-Charset** עבור קבוצת התווים.

לדוגמה, הנח שקבוצת התווים המקומית של הסוכן-משתמש היא ISO-8859-1, אולם הוא יכול להציג גם את קבוצות התווים ISO-8859-5 ו-ISO-8859-7. כאשר **TransMsg** פועל, ההודעה המוזנת מאוחסנת בתוכו ב-ISO-8859-1, קבוצת התווים המקומית. אם המשתמש בוחר ב-SO-8859-5 כקבוצת התווים הנשלחת, ההודעה מקודדת מחדש בדרכה ל-Java servlet **TransMsg**. ה-Servlet מקבל את ההודעה ב-ISO-8859-5, שולח אותה חזרה באותה קבוצת תווים, והיא נראית ומוצגת כקבוצת התווים ISO-8859-5.

להלן חלק ה-Java של היישום. אנו לא רק מחשבים באיזו קבוצת תווים להשתמש עבור גוף הודעת התגובה, אלא גם שולחים חזרה את הגדרת הכותרות **Accept-Charset** ו-**Content-Type** ואת המשתנים **message** ו-**charset**, כך שנראה מה מתרחש.

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.util.*;

public class TransMsg extends HttpServlet {

  static String EOL = "\r\n";

//***************************************************************
// doGet
//
// Respond to a GET request. Just call doPost. Included for Telnet testing.
//***************************************************************

  public void doGet (

    HttpServletRequest req,
    HttpServletResponse res )

    throws
    ServletException,
    IOException {

    doPost (req, res );
  }

  public void doPost (

    HttpServletRequest req,
    HttpServletResponse res)

    throws
    ServletException,
    IOException {

    String contType = req.getContentType ();
    String charSets = req.getHeader ( "Accept-Charset" );
    String message  = req.getParameter ( "message" );
```

```
    String charset  = req.getParameter ( "charset" );

    if ( charset == null )
      res.setContentType ( "application/vnd.wap.wml" );
    else
      res.setContentType ( "application/vnd.wap.wml; charset=" + charset );

    res.setHeader      ( "Cache-Control", "no-cache" );
    res.setHeader      ( "Content-location", "http://www.worldfaq.com/servlet/transMsg"
);

    ServletOutputStream out = res.getOutputStream();
    out.println (

      "<?xml version=\"1.0\"?>" +
      "<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML 1.1//EN\"" + EOL +
      "\"http://www.wapforum.org/DTD/wml_1.1.xml\">" + EOL +

      "<wml>" + EOL +
      "<card id=\"getContType\" >" + EOL +
      "<p>" + EOL +
      "AcceptCharsets: " + charSets + "<br/>" + EOL +
      "Content-type: " + contType + "<br/>" + EOL +
      "Message: " + message + EOL +
      "</p>" + EOL +
      "</card>" + EOL +
      "</wml>"
    );

}  }
```

הצגת עברית בדפי WML

כדי לאפשר הצגת עברית בדפי WML, יש להשתמש בתכונת encoding של XML בהקדמת החפיסה:

```
<?xml version="1.0" encoding="ISO-8859-8" ?>
```

לאחר הוספת שורה זו לראש חפיסת WML תוצג השפה העברית כראוי. בפרק 7 העוסק בפיתוח יישומי WAP נדגים את השימוש באלמנט זה.

מסקנות

תוכל להבין ביתר קלות נושאי WAP i18N, אם ברשותך ידע מסוים אודות כותרות HTTP. לצורך טיפול נכון בנושאי שפה, עליך להכיר את הכותרות **Accept-Language** ו-**Content-Language**. כדי לטפל נכון בקבוצות תווים, עליך להכיר את הכותרת **Accept-Charset** ואת התכונה **charset** של הכותרת **Content-Type**. מצויד בידע זה, תוכל בקלות ליצור יישום WAP שימושי ורב-לשוני.

פרק 7

יצירת יישומי WAP ו-ASP

פרק זה נכתב על ידי שרון טל.

עד כה עסקנו ביצירת דפי wml סטטיים. פרק זה יעסוק ביכולות הדינמיות של WAP ויכולות האינטגרציה שלו עם מסדי נתונים.

אופן יצירת דפי WML דינמיים דומה מאוד ליצירת דפי html דינמיים באמצעות שפות ASP ו- CGI.

באמצעות שימוש בדפי WML דינמיים ניתן לבצע:

- אינטגרציה עם בסיסי נתונים כגון SQL ו- ORACLE
- בניית פלטפורמות למסחר אלקטרוני.
- ניהול תוכן באופן דינמי.

בפרק זה נדגים בניית קטלוג ספרים WAP דינמי באמצעות טכנולוגיית ASP.

הפרק אינו עוסק בטכנולוגיית ASP, אלא בשילובו ביישומי WAP. להרחבת הידע בתחום ה- ASP ניתן להסתייע בספרי ASP של הוצאת **הוד-עמי**.

כיצד עובד ASP?

ASP היא טכנולוגיה המתבצעת על גבי שרת האינטרנט ומחזירה קוד WML לדפדפן הסלולרי. הדפדפן מציג את דפי WML הדינמיים, כשם שהוא מציג דפי WML סטטיים.

ASP נועד במקור להחזיר דפי HTML. כדי לגרום לקובץ ASP להחזיר לסוכן-משתמש (user agent) דפי WML, חובה להוסיף לכל קוד ASP את השורה הבאה בראש המסמך:

```
<% Response.ContentType = "text/vnd.wap.wml"%>
```

מבנה כללי: ASP משולב ב-WML

```
<% Response.ContentType = "text/vnd.wap.wml" %>
<?xml version="1.0" encoding="iso-8859-8"?>

<wml>

<%
ASP code
%>

</wml>
```

בניית קטלוג ספרים דינמי בשילוב ASP

הקוד הבא מדגים קטלוג ספרים המחולק לפי קטגוריות. עם טעינת הדף הראשי יוצגו בפני המשתמש רשימת הקטגוריות הקיימות בקטלוג. כל קטגוריה תוצג כהיפר קישור המוביל לדף שני המציג את רשימת הספרים השייכים לקטגוריה שנבחרה.

יישום זה כולל שלושה מרכיבים:

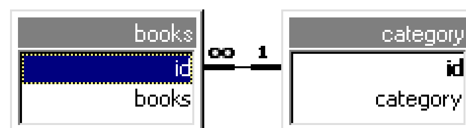
1. מסד נתונים מסוג Access.
2. קובץ ASP להצגת הקטגוריות מתוך מסד הנתונים.
3. קובץ ASP להצגת הספרים, בהתאם לקטגוריה שנבחרה על ידי המשתמש.

את שלושת הקבצים יש לשמור בשרת. לדוגמה:

C:\Inetpub\wwwroot\catalog

יצירת בסיס הנתונים (hodami.mdb)

ניצור בסיס נתונים פשוט המכיל שתי טבלאות: category ו- books, ונגדיר קשרי גומלין בין שדות ID (ראה תרשים 7.1).



תרשים 7.1

נזין את בסיס הנתונים באופן ידני, או על ידי יישום Web.

טבלה : books

books	id
פלאש 5	12
עיצוב ממשק האינטרנט	12
דרייקטור 8	12
פלאש 4	12
pain shop pro 6	12
photo shop 4	12
paint shop pro 5.x	12
XML	13
ASP	13
HTML 4	13
javaScript	13
DHTML	13
מבוא לתקשורת מחשבים	14
תקשורת מחשבים	14
MCSE	14
window me	15
Linux	15
המדריך השלם לטכנאי pc	16
	17

רשומה: 1 מתוך 21

טבלה : category

category	id
גרפיקה	12
אינטרנט	13
תקשורת ורשתות	14
מערכות הפעלה	15
טכנאי PC	16
שפות תכנות	17
בסיסי נתונים	18
מבחני הסמכה	19
מוצרים נוספים	20
	*

רשומה: 1 מתוך 9

תרשים 7.2

שליפת רשימת הקטגוריות (main.asp)

```
<%
 Response.Expires = -1
 Response.AddHeader "Pragma", "no-cache"
 Response.AddHeader "Cache-Control", "no-cache, must-revalidate"
 Response.ContentType = "text/vnd.wap.wml"

%>
<?xml version="1.0" encoding="iso-8859-8"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
 <!-- THIS IS THE MAIN CARD OF THE DECK -->
 <card title="הוד עמי">
  <%
   set c=server.createobject("adodb.connection")
   DSN = "DRIVER={Microsoft Access Driver (*.mdb)}; DBQ=" &
Server.Mappath("hodami.mdb") & ""
   c.Open DSN
   set r=server.createobject("adodb.recordset")
   r.activeconnection=c
   r.open "select * from category"
```

```
    do until r.eof
    response.write "<p align='right'><a href='result.asp?p=" &_
       r.fields("id") & "'>" & r.fields("category") &_
       "</a></p>"
    r.movenext
    loop
   %>
  </card>
</wml>
```

כך נראה הקובץ בטלפון של Nokia:



תרשים 7.3

הצגת רשימת הספרים בהתאם לקטגוריה הנבחרת (result.asp)

```
<% '**********************************************
   'גורם לדפדפן לטעון את הדף ישירות מהשרת ולא מהמטמון
   Response.Expires = -1
   Response.AddHeader "Pragma", "no-cache"
   Response.AddHeader "Cache-Control", "no-cache, must-revalidate"
   '**********************************************
   Response.ContentType = "text/vnd.wap.wml"

%>

<?xml version="1.0" encoding="ISO-8859-8" ?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">

<wml>
<!-- THIS IS THE MAIN CARD OF THE DECK -->
  <card id='result' title='ספרי הקטגוריה'>
    <%
      set c=server.createobject("adodb.connection")
      DSN = "DRIVER={Microsoft Access Driver (*.mdb)}; DBQ=" &_
            Server.Mappath("hodami.mdb") & ""
      c.Open DSN
      set r=server.createobject("adodb.recordset")
      r.activeconnection=c
      r.open "select * from books where id='"&request.querystring("p")&"'"
      do until r.eof
      response.write"<p align='right'>"&r.fields("books")&"</p>"
      r.movenext
      loop
      r.close
    %>
    <p align="center"><a href="main.asp">חזרה לתפריט</a></p>
  </card>
</wml>
```

פרק 8

Caching (מיטמון)

בפרק זה נרחיב מעט וננצל את יתרונותיהם של כמה מההיבטים המתקדמים יותר של WAP 1.1. נדריך אותך כיצד לשפר ביצועים, ולמזער את התנועה ברשת. לפני שניכנס לפרטים, להלן הקדמה קצרה אודות HTTP 1.1 כמתואר ב- RFC 2616. אם HTTP 1.1 מוכר לך, ובעיקר כותרות דרישה ותגובה, באפשרותך לדלג על סעיפים אלה בפרק.

הקדמה קצרה אודות HTTP 1.1

HTTP 1.1 הוא בעיקרו פרוטוקול מבוסס-טקסט, המגדיר את האינטראקציה בין ישויות באינטרנט, בין אם הם סוכני-משתמש, כגון דפדפנים תואמי WAP, או שרתי-תוכן. האינטראקציה ניתנת לסיווג לשתי קטגוריות: WAP proxy gateways דרישות לקוח משרתים ותגובות שרתים ללקוחות. אינטראקציה יחידה של סוכן-משתמש/שרת מורכבת מדרישה מהסוכן אל שרת, ובעקבותיה תגובה של השרת חזרה לסוכן.

הדרישות והתגובות, גם יחד, משתמשות בתבנית הודעות האינטרנט הכללית המתוארת ב- RFC 822.

Message Definition
zero or more message headers
CRLF
optional message body

Message Definition מציינת את סוג ההודעה הנשלחת. הן הדרישות והן התגובות יכולות להכיל כותרות הודעה שונות, המגדירות את האינטראקציה של הסוכן-משתמש והשרת. חלק ה-CRLF של ההודעה מפריד את הגדרות ההודעה מגוף ההודעה.

דרישות

Message Definition עבור דרישה נראית כך :

request-type URL HTTP/1.1

request-type (סוג דרישה) הוא אחד מהבאים :

- **OPTIONS**. מחזירה מידע אודות אפשרויות התקשורת הזמינים בין הדורש למגיב. הדבר מועיל לצורך הגדרת יכולות השרת.
- **GET**. מאחזרת את תוכן הקובץ, או את תוצאות ביצוע תוכנית, המזוהה על ידי URL. לדוגמה, השרת מחליט אם כתובת-המטרה היא קובץ סטטי או תוכנית, תוך שימוש בסיומת השם.
- **HEAD**. מאחזרת את אותו סוג מידע כמו **GET**, אך אינה מחזירה את גוף הידיעה של התגובה. מידע זה מועיל לצורך בדיקת קישורים, אימות, נגישות ושינויים אחרים.
- **POST**. שולחת את המידע שבגוף הידיעה אל ה-URL לצורך פעילויות שרת נוספות. **POST** משמשת בעיקר למסירת תכנים של טופס HTML, לצורך תהליך עיבוד כלשהו.
- **PUT**. שולחת את המידע שבגוף הידיעה ל-URL לאחסון. למרות שבדומה ל-**POST**, **PUT** היא בעלת הגדרה צרה, משתמשים בה לעיתים יותר רחוקות, והיא נתמכת פחות על ידי שרתים.
- **DELETE**. מוחקת את המקור המזוהה על ידי URL.
- **TRACE**. מפעילה מרחוק, loopback של שכבת-יישום על הודעת הדרישה. הסוכן-משתמש השולח הודעה זו ומקבל חזרה את אותה הודעה, בלוויית מידע נוסף אודות הנתיב שההודעה עברה בו דרך האינטרנט.

הדרישות השימושיות ביותר בעת יצירת יישום WAP, ושאליהן עלינו לשים לב בעיקר, הן **GET** ו-**POST**.

להלן דרישת GET עבור www.worldfaq.com/wml/wfaq/wml, שנוצרה על ידי המדמה Nokia. למדמה יש שורת פקודה שבה ביכולתך להזין URL לצורך קריאה והפעלה. כאשר אתה מזין :

www.worldfaq.com/wfaq1.wml

בשורת הפקודה, ומקיש Enter, דרישת ה-**GET** הנשלחת לשרת worldfaq היא כדלהלן :

GET www.worldfaq.com/wfaq1.wml HTTP/1.1
accept-charset: UTF-8
accept-language: en
accept: text/vnd.wap.wml, */*, image/bmp, text/html
user-agent: UP.Browser/3.1-UPG1 UP.Link/3.2
host: www.worldfaq.com

כותרות HTTP מסומנות בהבלטה. כדי להקל על הבנת הדוגמה זו ובהירותה, הסרנו מספר סוגי **accept** וכמה כותרות אפיון. להלן תיאור קצר של הכותרות:

- **accept-charset.** קבוצה (או קבוצות) שהסוכן-משתמש יכול להציג.
- **accept-language.** השפה שבשימושו הנוכחי של הסוכן-משתמש.
- **accept.** סוג מסמך MIME שהסוכן-משתמש יכול להשיג בהצלחה.
- **user-agent.** השם שניתן על ידי היצרן לסוכן-משתמש.
- **host.** ה-domain אליו נשלחת הדרישה.

לפירוט רב יותר אודות **accept-charset** ו-**accept-language**, ראה פרק 6.

תגובות

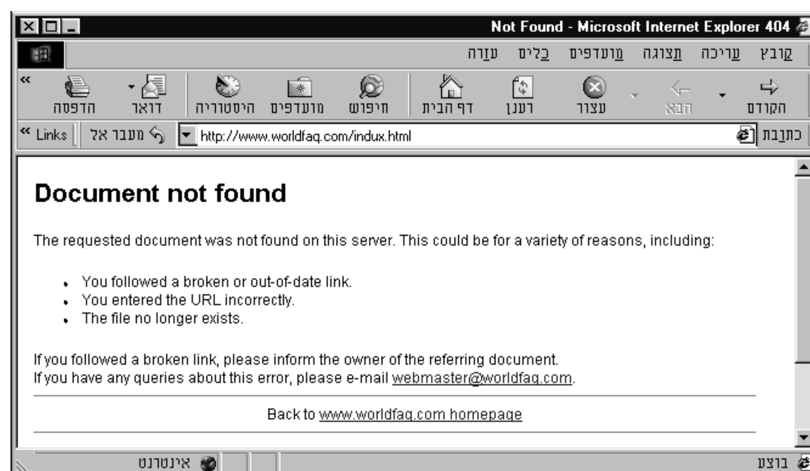
ה-**Message Definition** של התגובות נראית כך:

HTTP/1.1 Status-code Status-description

RFC 2616 מגדיר כמעט 40 סוגים של קוד-מצב המסווגים לחמש קבוצות. הנפוצות שבהן:

200 OK
401 Unauthorized
404 Not Found

ודאי ראית את דפדפן ה-Web שלך מציג את תגובה מספר 404. היא עשויה להשתרע בטווח שבין הודעה פשוטה בת שורה אחת ועד לתגובה יותר מפורטת כמו זו המוצגת בתרשים 8.1.



תרשים 8.1 תגובת 404 טיפוסית

כבר קיימנו דיון בתגובות, הודעות וכותרות HTTP. לדוגמה, אמרנו שהתגובה ל:

GET www.worldfaq.com/wfaq1.wml HTTP/1.1

היא החפיסה המאוחסנת בקובץ wfaq1.wml שעל שרת WorldFAQ. למעשה, התגובה נראית דומה לזו:

```
HTTP/1.1 200 OK
Server: Zeus/3.1
Date: Tue, 20 Jul 1999 21:38:04 GMT
Connection: Keep-Alive
Content-Length: 3343
content-Type: text/vnd.wap.wml
Last-Modifed: Set, 17 Jul 1999 21:19:02 GMT

<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
<http://www.wapforum.org/DTD/wml_1.1.xml">
'
' The rest of the deck stored in wfaq1.wml
'
```

תגובה זו מכילה את מצב התגובה, את הקוד הנומרי ואת תוכן הודעת הטקסט, ואחריהם קבוצה של כותרות, שבעקבותיהן החזרת גררה, שורה חדשה וגוף ההודעה. גוף ההודעה במקרה זה הוא קובץ קוד המקור המוצב ב-www.worldfaq.com/wfaq1.wml.

להלן תיאור קצר של כותרות התגובה:

- **Server**. שרת ה-Web שמסר את התגובה.
- **Date**. התאריך והשעה של שליחת ההודעה.
- **Connection**. הוראה לסוכן-משתמש לשמור על קשר זה כאקטיבי.
- **Content-Length**. אורך הודעת התגובה, המתחיל בתו "<" הראשון של חפיסת WML.
- **Content-Type**. סוג תוכן ה-MIME של התגובה.
- **Last-Modified**. התאריך שבו הקובץ המכיל את חפיסת התגובה, עודכן לאחרונה.

כאשר הסוכן-משתמש מקבל את התגובה, הוא מפענח את ה-status ואת הכותרות, ומחליט מה לעשות עם ההודעה. בדרך כלל, במקרה של תגובת OK, הסוכן-משתמש מציג את גוף ההודעה על מסך ההתקן ששלח את הדרישה. עבור דפדפן Web של מחשב נישא, גוף ההודעה (יש לקוות) הוא ב-HTML; ועבור דפדפן WAP, ב-WML.

HTTP הוא פרוטוקול מרובה מילים. אפילו דרישות ותגובות פשוטות, למעשה חסרות נתונים, יכולות להיות בנות מאות תווים. WAP מתמודד עם בעיה זו על ידי הדרישה שכל יישומי WAP ישתמשו ב- WAP gateway. בין יתר תפקידיו, ממיר WAP gateway הודעות HTTP 1.1, דרישות ותגובות כאחד, להודעות **W**ireless **S**ession **P**rotocol (WSP). WSP המהווה פרוטוקול בינארי קומפקטי, הוא תואם HTTP 1.1. פעולותיו כוללות לקיחה של דרישה או תגובה כדוגמת אלו שראינו עד כה, ניתוחה והמרתה ל-bits בצורה הקומפקטית ביותר האפשרית.

ברור ש-HTTP 1.1 הוא פרוטוקול הרבה יותר מורכב מההקדמה הקצרה שניתנה על ידנו. RFC 2616 מספק מיפרט ארוך ומפורט, המגדיר כיצד סוכני-משתמש ושרתי-תוכן יוצרים אינטראקציה ביניהם. אין בכוונתנו לתאר בפירוט את כל תבניות ההודעה והכותרת של HTTP 1.1. אנו רוצים שתזכור נקודת מפתח אחת: מלבד הדרישות **GET**, **POST** ותוכן סטטי או דינמי, חומר רב נוסף, מועבר הלוך ושוב בין הסוכן-משתמש לשרת-התוכן. קיימות גם כותרות דרישה ותגובה, שלהן השפעה משמעותית על אופן הפעלתם וביצועם של יישומי WAP.

כפי שתיווכח במהרה, הבנת מספר כותרות-מפתח של HTTP היא הסוד למזעור התנועה ברשתות הסלולריות בסביבת WAP.

מיטמון (Caching)

לפי RFC 2616, מטמון הוא "מחסן מקומי של הודעות תגובה, ותת-מערכת השולטת בהודעות, באחזורן ובמחיקתן. מטמון מאכלס תגובות הניתנות למיטמון, לצורך הפחתת זמן התגובה והקטנת צריכת רוחב-פס של הרשת עבור דרישות עתידיות שוות ערך". ב- RFC 2616 קיים פירוט נרחב אודות מיטמון. למזלנו אנו יכולים להתעלם מרובו.

בעת כתיבת יישומי WAP, שאיפתך צריכה להיות להפחית ככל האפשר את תנועת ההודעות ברשת. הדרך לעשות זאת היא למטמן ככל שניתן, והשגה מן המטמון לעיתים קרובות ככל האפשר. על כל התהליך להיות שקוף למשתמש. למרבה המזל, למרות שאין הדבר נדרש, רוב התקני WAP הינם בעלי יכולת מיטמון מסוימת, וכברירת מחדל הם מנסים למטב יכולת זאת. אלא אם כן קיימת הוראה אחרת, התגובות לכל דרישות URL מוטמנות.

כאשר סוכן-משתמש WAP מטמין תגובה. הוא מטמין כמעט את כל המרכיבים: את ה-URL, טקסט התגובה, הכותרות השייכות למיטמון ומידע לצורך אימות התגובה (למידע נוסף, ראה סעיף "תקפות ומחסנית ההיסטוריה" בהמשך הפרק). כל פריט מוטמן הוא בעל זיהוי ייחודי, באמצעות קבוצה שלמה של מרכיבי URL: domain name, נתיב, פרוטוקול, פרמטרים, מספר יציאה ועוד.

קיימות שתי כותרות HTTP היכולות לשמש אותך לשליטה בחפיסות WML יחידות ובפונקציות WMLScript. הכותרת החשובה למטרתנו היא **Cache-Control**. כותרת זו היא מנגנון מיוחד, שבאמצעותו ניתן לשלוט בכל הישויות המוטמנות, לכל אורך שרשרת הדרישה/תגובה. כל מנגנון המיטמון חייב לציית לכותרות אלו. כותרות **Cache-Control** דורסות, בדרך כלל, התנהגות ברירת מחדל של מטמון-התקן. הן חייבות להיות מועברות ללא שינוי דרך כל ה-proxies וה-Gateways בשרשרת ההודעות.

- **Cache-Control: no-cache** מציינת שה-URL לא יבצע מיטמון על ידי הסוכן-משתמש, או על ידי כל שרת הנמצא בין שרת-התוכן והסוכן-משתמש.
- **Cache-Control: max-age=<seconds>** מאפשרת לך להגדיר את אורך הזמן ש-URL אמור לשהות במטמון ההתקן. כאשר הישות עברה את הזמן המקסימלי בשניות, היא אמורה להיות מוסרת.
- **Expires: <date>** מאפשרת לך להגדיר את התאריך שאחריו URL אמור להיות מוסר מהמטמון. תבנית התאריך/שעה עבור כל כותרות HTTP מוגדרת פורמלית ב- RFC 1123. כותרת תפוגה טיפוסית נראית כך:

```
Expires: Thu, 29 Apr 2002 19:47:52 GMT
```

בעת כתיבת יישומי WAP, תוכל להניח כי הסוכן-משתמש יעשה כל שביכולתו כדי למקסם את המיטמון ולמזער פניות חוזרות לשרת-תוכן. קיימים ארבעה מקרי מיטמון שונים שעליך לשקול כדי לדרוס את מיטמון-ברירת המחדל עבור היישום שלך.

מיטמון URL מתמיד

בדרך כלל, סוכן-משתמש WAP שומר URL במטמון זמן רב ככל האפשר והוא נפטר מהכתובת רק כשנחוץ. דפדפן Phone.com מגדיר "זמן רב ככל האפשר" כ- 30 יום בערך – חלון זמן רחב מאוד, אם אתה משתמש בהתקן בקביעות. לעומת זאת, ישנם מקרים בהם תרצה למטמן פריט לזמן רב יותר. הבה נאמר, לדוגמה, שברצונך להשתמש בקובץ גרפי קטן המכיל את לוגו החברה שלך כמסך רקע לשירות. אתה יודע שהלוגו שלך לא ישתנה בזמן הקרוב. מה הטעם להכריח את הסוכן-משתמש לטעון מחדש את הקובץ כל 30 יום, מדוע לא למטמן אותו לשנה או אפילו יותר?

קיימות שתי דרכים לבצע זאת. הדרך הראשונה היא להגדיר תאריך **Expires** (תפוגה) הרחק בעתיד:

```
Expires: Mon, 01 Jan 2003 00:00:00 GMT
```

הדרך השנייה היא להגדיר ערך **max-age** גדול בכותרת **Cache-Control**:

```
Cache-Control: max-age=31536000
```

כאשר ערכו המקסימלי של integer של סוכן-משתמש הוא 2,147,483,647 וישנן 84,000 שניות ביממה, תוכל להגדיר זמן של עד 24,000 ימים. זמן שהוא ודאי ארוך מאורך חייו של התקן WAP שלך.

מיטמון URL לזמן מוגדר

סביר להניח שהמקרה הנפוץ יותר בו נתקלת הוא מיטמון URL לזמן מוגדר. לדוגמה, ייתכן מצב שבו עומד לרשותך שירות הצגת שערי מניות המתחדש כל 15 דקות. אם המשתמש מרענן את המידע בנקודה מסוימת. ושוב, חמש דקות מאוחר יותר, אתה עשוי לרצות להציג מספר חדש. הדבר תלוי בשאלה האם מספר חדש כזה זמין.

תוכל לחשב את מספר השניות בין הבקשה הנוכחית והגעת המספרים החדשים. חישוב זה תלוי במספר גורמים פנימיים וחיצוניים המשפיעים על זמן התגובה. תוכל לאחסן מידע זה בכותרת **Cache-Control:max-age=<seconds>**, ואז, אם המשתמש דורש ערך חדש לאותו שדה לפני שפג תוקף **max-age**, הנתונים המוטמנים הם שמוצגים. אחרת, המידע נדרש מחדש משרת-התוכן.

הדרך השנייה שביכולתך לעשות שימוש במידע זה, היא על ידי ציון בכותרת **Expires** את התאריך והזמן שבהם פג תוקף המידע. הסוכן-משתמש יכול להשתמש במידע זה כדי לחשב מתי לפלוט את ה-URL מהמטמון ולהשיג ציטטה חדשה.

לדוגמה, כדי למטמן תגובה למשך חמש דקות, בהנחה שהתגובה נוצרה בשעה GMT10 לפני הצהריים ב- 10 בינואר 2000, ביכולתך להשתמש באחת מהכותרות הבאות :

Expires: Mon, 10 Jan 2000 10:05:00 GMT
Cache-Control: max-age=300

ביטול אפשרות מיטמון URL

בשירותים תנודתיים בעלי שינויים מהירים, בהם יש להתמיד ולחדש תוכן מסמך משרת-התוכן, עליך לבטל את המיטמון לחלוטין. הדבר יכול להידרש, לדוגמה, עבור שירות המספק זרם נתוני מניות בזמן-אמת, והמתעדכן בצורה שוטפת במהלך היום. כאשר אתה משיג את המידע המפורט אודות מקום, הוא כולל את התאריך והזמן הנוכחיים, אותם תמיד יש להשיג מחדש.

ביכולתך לבטל את המיטמון תוך שימוש בכותרת **Cache-Control: no-cache**, כמו כן תוכל להשתמש ב-**Cache-Control: max-age=0** או בכותרת **Expires** בעלת תאריך שפג תוקפו. אולם, האפשרויות השנייה והשלישית אינן בחירות טובות. ראשית, מכיון שהן דורשות יותר עבודה מצדו של הסוכן-משתמש : לא רק שעליו לחפש תחילה את ה-URL שבמטמון, אלא שאם הסוכן-משתמש מצא את ה-URL, הוא חייב לחשב את גילו. שנית, הדבר עומד בסתירה לכוונתך האמיתית. שלישית, הן דורשות מעט יותר bits עבור הודעות. ביכולתך לבטל את פעולות החישוב הנוספות על ידי שימוש ב-**Cache-Control: no-cache**.

שלוש הכותרות נותנות לך תוצאה סופית זהה, אם התגובה נוצרה בשעה GTM 10 לפני הצהריים ב- 10 בינואר 2000.

Cache-Control: no-cache
Cache-Control: max-age=0
Expires: Sun, 10 Jan 2000 09:00:00 GMT

תקפות ומחסנית ההיסטוריה

בנוסף למיטמון, HTTP 1.1 מבצעת בדיקת תקפות. תקפות היא התהליך לקביעת תקפותו של ערך מטמון. היא קובעת האם פג תוקפו של הערך, או לא. בדיקת התקפות מסבכת במעט את מודל סוכן-משתמש WAP, עקב הימצאותה האפשרית של מחסנית היסטוריה.

סטנדרט WAP מעודד את כל יצרני ההתקנים לכלול מחסנית היסטוריה של 10 פריטים לפחות. כל פעם ש-URL מושג תוך שימוש באלמנט <**go**>, הוא מאוחסן במחסנית. כל פעם שה-URL מושג תוך שימוש באלמנט <**prev**>, או על ידי מנגנון החיפוש לאחור של התקן WAP, ה-URL "מוקפץ" החוצה ממחסנית ההיסטוריה. הסוכן-משתמש יכול להשתמש ב-URLs אלה כדי לחפש אחר תגובות מוטמנות לפרטיהן.

כדי לוודא את פעולתו התקינה של היישום שלך, עליך להבין כיצד המחסנית והמטמון פועלים יחד. הכלל הבסיסי הוא שתקפותן של כל ההתייחסויות קדימה (forward references) נבדקת, ואילו תקפותן של כל ההתייחסויות אחורה אינה נבדקת.

בהתייחסויות קדימה, הסוכן-משתמש WAP, תמיד בודק תחילה האם ה-URL נמצא במטמון. אם לא, הוא יוצר דרישה עבורו. כאשר הוא מקבל את התגובה הוא מטמין אותה (אם הוא מורשה לעשות זאת), ודוחף את ה-URL למחסנית ההיסטוריה. אם ה-URL כבר נמצא במטמון, הסוכן-משתמש מוודא שה-URL ותוכנו עדיין תקפים. אם הוא תקף, הפריט המוטמן משמש כתגובה.

אם ה-URL אינו תקף יותר מכיון, לדוגמה, שזמן התפוגה שלו עבר, הסוכן-משתמש דורש שוב את ה-URL מהשרת. ערך המיטמון הישן מוסר, והתגובה החדשה מוכנסת למטמון (אם יש לסוכן-משתמש הרשאה). ה-URL מאוחסן במחסנית ההיסטוריה, והתגובה החדשה מטופלת על ידי הסוכן-משתמש.

באשר להתייחסות אחורה, התגובה הרגילה של הסוכן-משתמש דומה למה שקורה כשאתה לוחץ על לחצן Back שבדפדפן המחשב הרגיל. פעולת <**prev**> היא דרישה לתגובה ההיסטורית האחרונה עבור URL. המשתמש רוצה את מה שכבר נמצא במטמון, כך שאין טעם לבדוק האם הוא עדיין תקף (במילים אחרות, אין מקום לבדיקת תקפות).

לעומת זאת, קיים מקרה, בו תבקש פעולת <**prev**> שתבצע בדיקת תקפות חוזרת לתגובה, כדי לוודא שהיא עדיין טרייה. למרבה המזל, קיימת הגדרת כותרת **Cache-Control** המבצעת זאת. השתמש ב:

Cache-Control: must-revalidate

כדי לכפות על סוכן-משתמש WAP לוודא תקפות של ערך במחסנית ההיסטוריה בעת ביצוע דפדוף אחורנית. וידוא תקפות הערך, אין פירושו שה-URL מושג מחדש – הוא מושג מחדש רק אם אינו תקף יותר. אם הוא עדיין תקף, נעשה שימוש בתגובה המוטמנת.

כדי לכפות טעינה מחדש מתמדת בהתייחסות <**prev**>, עליך להשתמש ב:

Cache-Control: must-revalidate , no-cache

אם ברצונך לכפות טעינה מחדש בכל התייחסות <**prev**> שהיא בת יותר מחמש דקות, השתמש ב:

Cache-Control: must-revalidate, max-age=300

לסיכום, השתמש ב-**Cache-Control: must-revalidate** כדי לשלוט בהתייחסות אחורה של מיטמון, והוסף כל כותרת הדרושה לך כדי לבטא קריטריון.

ישנו מיגוון שיטות אימות הקשורות להתיישנות מיטמון, מעבר לאלו שדנו בהן. דיון מפורט בכל שיטות האימות, הוא מעבר לטווח ספר זה. לפרטים נוספים ראה RFC 2616.

כותרות HTTP לעומת אלמנטי META

ביכולתך לנהל מיטמון ולוודא תקפות, תוך שימוש בכותרות HTTP. שאלה הגיונית היא כיצד תשלוט בכותרות HTTP, הכלולות בתגובות הנשלחות משרת-התוכן שלך, חזרה אל הסוכן-משתמש WAP?

כאשר שרת יוצר תגובה לדרישת סוכן-משתמש, הוא אוטומטית מצרף לתחילת התגובה, כל כותרת שלדעתו מתאימה. אם אתה יוצר חפיסה סטטית המוחזרת בגוף התגובה, אין לך כל דרך להגדיר ערך כלשהו של כותרת HTTP. הדבר נכון גם אם אתה יוצר תוכן דינמי, תוך שימוש בטכניקה שאינה מספקת כל אמצעים להגדרת כותרות.

אלמנט WML **http-equiv <meta>** מעוצב כדי להתמודד עם מגבלה זאת, כך שתוכל להגדיר ערכי כותרת תגובה מתוך מסמכים סטטיים ודינמיים גם יחד. כאשר שרת מחזיר מסמך WAP כתגובה לדרישה, WAP gateway סורק את גוף ההודעה, תוך חיפוש רצף תווים כדוגמת אלה:

```
<meta http-equiv="Cache-Control" content="no-cache" />
```

אם הוא מוצא את רצף התווים מהסוג המבוקש, הוא מרחיב את תוכן http-equiv, וממיר אותו ל-WSP שווה הערך לכותרת **Cache-Control** של HTTP. כותרת זו מועברת הלאה לסוכן-משתמש, כך שהוא יודע שאין למטמן את התגובה. כך תוכל להגיע לאותה תוצאה, כאילו יצרת את הכותרת והצבת אותה ישירות בחלק הכותרת של התגובה.

להלן כמה מהכותרות שכבר ראינו, וה-<**meta**> שווי הערך שלהן:

```
Expires:  Mon,  10  Jan  2000  10:05:00 GMT
Cache-Control: max-age=300
Cache-Control:  no-cache
```

```
<meta http-equiv="Expires" content=" Mon,  10  Jan  2000  10:05:00 GMT" />
<meta http-equiv="Cache-Control" content="max-age=300" />
<meta http-equiv="Cache-Control" content="no-cache" />
```

אם אין לך דרך לשליטה ישירה בכותרות שבתגובת HTTP, אלמנט <**meta**> הוא הכרחי, אך הוא גורם למעט עבודה נוספת. ראשית, עליך לוודא שהוספת את אלמנט <**meta**> לחפיסת התגובה שלך. שנית, הוא דורש עבודה נוספת בחלק WAP gateway. הסיבה השלישית היא היסטורית – לא כל סוכני-המשתמש והשרתים מכבדים אלמנטי <**meta**>. אם הדבר אפשרי, עליך להשתמש במקום זאת בכותרות HTTP לשליטה במיטמון של סוכני-משתמש.

למרבה המזל, הטכנולוגיות של מספר שרתי-תוכן מספקות שיטות יצור ישיר של כותרות תגובה של HTTP. כך הוא המצב עם Java servlets: ישנם APIs המיוחדים לכתיבת כותרות תגובה.

בדיקה עם Telnet

Telnet היא טכניקה בה ניתן להשתמש לצורך ממשק עם שרתי Web, ולהבנת המתרחש כאשר אתה מפעיל תוכניות מבוססות-שרת. לרוב מערכות ההפעלה יש Telnet clients חופשיים. Telnet היא פקודה סטנדרטית של מערכת Unix. ביכולתך לגשת לתוכנת Telnet כתוכנת שורת-פקודה ב-Windows. קיימות מספר תוכנות חופשיות של Telnet clients עבור Macintosh ו-OS.

פרוטוקול Telnet, שלא נדון בו כאן, מאפשר לך ליצור קשר TCP/IP לשרת Web, בדרך כלל ביציאה 80 – יציאת ברירת המחדל של HTTP. משם ביכולתך לשלוח פקודת HTTP על כל פרטיה, כולל כותרות, לשרת. השרת מבצע את הפקודה, ומחזיר אליך את התגובה. הדבר המעניין הוא שאתה מקבל את כל התגובה – שורת מצב, כותרות וגוף הודעה (אם קיים כזה). זוהי דרך מצוינת להבין כיצד פועל HTTP ואת תפקידו של שרת ה-Web.

נספח א

WML Elements

DENTRY	Data-entry elements: <input>, <select>, and <fieldset>.
DOCHOICES	Valid <do> choices: accept, prev, help, reset, options, delete, and unknown.
FMTTEXT	Elements for formatting text: <b>, <big>, <em>, <I>, <small>, <strong>, <u>.
ID	An XML-compatible name that uniquely identifies an element within a document.
IEVENTS	Card-level intrinsic events: onenterforward, onenterbackward, and Ontimer.
LENGTH	An integer to indicate length in pixels, or an integer plus a percent sign to indicate length as a percentage of the screen width.
NAME	A valid XML name-letters, digits, and underscore character.
NUMBER	A valid integer greater than or equal to zero.
STRING	Single-line Unicode 2.0 text that is not parsed.
TEXT	Multiline Unicode text that is not parsed.
VDATA	A STRING with possible variable references.
URL	An absolute or relative URI, URL, or URN, possibly containing variable references.

	ATTRIBUTES	ELEMENTS
a	href=URL id=IO titte=DATA class=STRING xml:lang=NAME	TEXT br img
access	domain=STRING id=ID path=STRING class=STRING	
anchor	title=VDATA id=ID xml:lang=NAME class=STRING	TEXT go img br prev refresh
b	xml:lang=NAME class=STRING id=ID	TEXT mg table FMTTEXT anchor br a
big	xml:lang=NAME class=STRING id=ID	TEXT img table FMTTEXT anchor a br
br	xml:lang=NAME class=STRING id=ID	

	ATTRIBUTES		ELEMENTS		
card	title=VDATA	onenterbackward=URL	onevent (onenterforward I onenterbackward I ontimer)		
	newcontext=true false	ontimer=UFIL	do	p	
	ordered=true false	id=ID	timer		
	xml:lang=NAME	class=STRING			
	onenterforward=URL				
do	type=DOCHOICES	xml:larrg=NAME	go	noop	
	label=VDATA	id=ID	prev	refresh	
	name=NAME	class=STRING			
	optional=true | false				
em	xml:lang=NAME	class=STRING	TEXT	img	table
	id=ID		FMTTEXT	anchor	a
tleldset	title=VDATA	id=ID	TEXT	img	table
	xml;lang=NAME	ciass=STRING	FMTTEXT	anchor	do
			DENTRY	a	

	ATTRIBUTES		ELEMENTS		
go	hret=URL sendreterer=true | false method= post | get	accept-charset=STRING id=ID class=STRING	postfield setvar		
head	id=ID	class=STRING	access	mets	
i	xml:lang=NAME id=ID	class=STRING	TEXT FMTTEXT br	mg anchor	table a
img	alt=VDA TA arc=URL localarc=VDATA vspace=LENGTH (0) hspace=LENGTH (0) align=top|middle|bottom	height=LENGTH width=LENGTH xml:lang=NAME id=ID class=STRING			

	ATTRIBUTES		ELEMENTS	
Input	name=NAME	maxlength=NUMBER		
	type=text password	tabindex=NLJMBER		
	value=VDATA	fitle=VDATA		
	format=STRING	xml:lang=NAME		
	emptyok=true | false	id=ID		
	size=NUMBER	class=STRING		
mete	http-equiv=STRING	scheme=STRING		
	name=STRING	id=ID		
	forua=true I false	class=STRING		
	content=STRING			
noop	id=ID	class=STRING		
oneVeflt	type=IEVENTS	class=STRING	go	noop
	id=ID		prev	refresh
optgroup	title=VDATA	id=ID	optgroup	
	xml:lang=NAME	class=STRING	option	

	ATTRIBUTES		ELEMENTS		
option	value=VOATA title=VDATA onpick=URL	xml:lang=NAME id=ID class=STRING	TEXT onevent (onpick)		
p	align=left I right I center mode=wrap I nowrap xml:lang =NAME	id=ID class=STRING	TEXT FMTTEXT DENTRY	img anchor a	table do br
postfleld	name=VDATA values VDA TA	id=iD class=STRING			
prev	id=ID	class=STRING	setvar		
refresh	id=ID	class=STRING	setvar		
select	title=VDATA name=NAME value=VDATA iname=NAME value=VDATA	multiple=true I false tabindex=NUMEER xml:lang=NAME id=ID class=STRING	optgroup option		
setvar	name=VDATA value=VDATA	id=ID class=STRING			

	ATTRIBUTES	ELEMENTS
small	xml:lang=NAME class=STRING id=ID	TEXT img table FMTTEXT anchor br a
strong	xml:lang=NAME class=STRING id=ID	TEXT mg table FMTTEXT anchor br a
table	title=VDATA xml:lang=NAME align=STRING Id=ID columns=NUMBER class=STRING	tr
Id	xml:lang=NAME class=STRING id=ID	TEXT img br FMTTEXT anchor a
template	id=ID class=STRING onenterforward onenterbackward ontimer	do onevent (onenterforward I onenterbackward I ontimer)
timer	name=NAME id=ID value=VDATA class=STRING	

	ATTRIBUTES		ELEMENTS		
tr	id=ID	class=STRING	Id		
u	xml:lang=NAME id=IO	class=STRING	TEXT FMTTEXT br	img anchor	table a
wml	xml:lang=NAME id=ID	class=STRING	head template	card	

נספח ב

WML Elements Cross Reference

Check the lefthand column and read across horizontally.

	a	access	anchor	b	big	br	card	do	em	fieldset	go	head	i	img	input	meta	noop	onevent	optgroup	option	p	postfield	prev	refresh	select	setvar	small	strong	table	td	template	TEXT	timer	tr	u	wml
a						x								x																		x				
access																																				
anchor						x					x			x									x	x								x				
b	x		x	x	x	x			x				x	x													x	x	x			x			x	
big	x		x	x	x	x			x				x	x													x	x	x			x			x	
br																																				
card								x										x			x												x			
do											x						x						x	x												
em	x		x	x	x	x			x				x	x													x	x	x			x			x	
fieldset	x		x	x	x	x		x	x	x			x	x	x										x		x	x	x			x			x	
go																						x				x										
head		x														x																				
i	x		x	x	x	x			x				x	x													x	x	x			x			x	
img																																				
input																																				
meta																																				
noop																																				
onevent																																				
optgroup																			x	x																
option																		x														x				
p	x		x	x	x	x		x	x	x			x	x	x										x		x	x	x			x			x	
postfield																																				
prev																										x										
refresh																										x										
select																			x	x																
setvar																																				
small	x		x	x	x	x			x				x	x													x	x	x			x			x	
strong	x		x	x	x	x			x				x	x													x	x	x			x			x	
table																																		x		
td	x		x	x	x	x			x				x	x													x	x				x			x	
template								x										x																		
timer																																				
tr																														x						
u	x		x	x	x	x			x				x	x													x	x	x			x			x	
wml							x					x																			x					

Text Any valid Unicode string or WAP-defined character entity:

quote	"	quotation mark
amp	&	ampersand
apos	'	apostrophe
It	<	less than
gt	>	greater than
nbsp		nonbreaking space
shy		soft hyphen (discretionary hyphen)

נספח ג

WML Attributes Cross Reference

Check the top row and read down verticaly.

	a	access	anchor	b	big	br	card	do	em	fieldset	go	head	i	img	input	meta	noop	onevent	optgroup	option	p	postfield	prev	refresh	select	setvar	small	strong	table	td	template	timer	tr	u	wml	do
accept-charset=STRING											x																									
align=left | right | center																					x															
align=STRING																													x							
align=top | middle | bottom														x																						
alt=VDATA														•																						
class=STRING	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
columns=NUMBER																													•							
content=STRING																•																				
domain=STRING		x																																		
emptyok=true | false															x																					
format=STRING															x																					
forua=true | false																x																				
height=LENGTH																																				
href=URL	•										•																									
hspace=LENGTH (0)														x																						
http-equiv=STRING																x																				
id=ID	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
iname=NAME																									x											
ivalue=VDATA																									x											
label=VDATA								x																												
localsrc=VDATA														x																						
maxlength=NUMBER															x																					
method=post | get											x																									
mode=wrap | nowrap																					x															
multiple=true | false																									x											
name=NAME								x								x																x				
name=STRING																																				
name=VDATA															•							•			x	•										
newcontext=true | false							x																													
onenterbackward=URL							x																								x					
onenterforward=URL							x																								x					
onpick=URL																				x																
ontimer=URL							x																								x					
optional=true | false								x																												
ordered=true | false							x																													
path=STRING		x																																		
scheme=STRING																x																				
sendreferer=true | false											x																									
size=NUMBER															x																					
src=URL														•																						
tabindex=NUMBER															x										x											
title=VDATA	x		x				x			x					x				x	x					x											
type=DOCHOICES								•																												
type=IEVENTS																		•																		
type=onpick																		•																		

	a	access	anchor	b	big	br	card	do	em	fieldset	go	head	i	img	input	meta	noop	onevent
type=text | password															x			
value=VDATA															x			
vspace=LENGTH (0)														x				
width=LENGTH														x				
xml:lang=NAME	x		x	x	x	x	x		x	x		x	x	x	x			

	optgroup	option	p	postfield	prev	refresh	select	setvar	small	strong	table	td	template	timer	tr	u	wml	do
type=text | password																		
value=VDATA		x		•			x	•						•				
vspace=LENGTH (0)																		
width=LENGTH																		
xml:lang=NAME	x	x	x				x		x	x	x	x				x	x	x

*	Required attribute.
DENTRY	Data entry elements: <input>, <select> and <fieldset>.
DOCHOICES	Valid <do> choices: accept, prev, help, reset, options, delete and unknown.
FMTTEXT	Elements for formatting text: <b>, <big>, <em>, <i>, <small>, <strong>, <u>.
ID	An XML-compatible name that uniquely identifies an element within a document.
IEVENTS	Card-level intrinsic events: onenterforward, onenterbackward and ontimer
LENGTH	An integer to indicate lengh in pixels, or an integer plus a percent sign to indicate lengh as a percentage of the screen width.
NAME	A valid XML Name-letters, digits and underscore caracter.
NUMBER	A valid integer greater than or equal to zero.
STRING	Single-line unicode 2.0 text that is not parsed.
TEXT	Multiline unicode text that is not parsed.
VDATA	A STRING with possible variable references.
URL	An absolute or relative URI, URL or URN, possibly containing variable references.

נספח ד

WMLScript Library Functions

Lang

Arithmetic

abs (num)	Return the absolute value of num.
max (num1, num2)	Return the maximum of num1 and num2.
min (num1, num2)	Return the minimum of num1 and num2.

Conversion

isFloat (value)	Return true if value can be converted to float using parseFloat.
isInt (value)	Return true if value can be converted to integer using parseInt.
parseFloat (str)	Return the floating-point equivalent of str.
parseInt (str)	Return the Integer equivalent of str.

Environment

characterSet ()	Return the IANA character set value.
float ()	Return true if floating-point nums are supported.
maxInt ()	Return the maximum integer.

minInt ()	Return the minimum integer.
Flow Control	
abort (str)	Terminate, returning str.
exit (value)	Terminate, returning value.
Random Number	
random (int)	Return a random integer between zero and int.
seed (int)	Initialize the random num generator using int.

Float

Environment	
maxFloat ()	Return the maximum positive floating-point value.
minFloat ()	Return the smallest positive floating-point value.
Arithmetic	
ceil (num)	Return the smallest integer not less than num.
floor (num)	Return the greatest integer not greater than num.
int (num)	Return the integer part of num.
pow (num1, num2)	Return num1 to the num2 power.
round (num)	Return the integer closest to num.
sqrt (num)	Return the square root of num.

String

Basic	
charAt (str, num)	Return the str character at location num.
compare (str1, str2)	Lexical comparison of str1 and str2.

IsEmpty (str)	Return true if str's length is zero.
length (str)	Return the length of str.
squeeze (str)	Remove all consecutive white space from str.
trim (str)	Remove all leading and trailing white space from str.
Substring	
subStr (str, startNum, lenNum)	Return from str the substring starting at startNum and lenNum long.
find (str, SubStr)	Return the Index of the first location of subStr in str.
replace (str, oldStr, newStr)	Replace all oldStrs in str with newStr.
Element	
elementAt (str, num, sepStr)	Return the numth element in str that is separated by sepStr.
elements (str, sepStr)	Return the number of substrings in str that are separate by sepStr.
insertAt (str, elemStr, num, sepStr)	Insert elemStr in str at the numth element that is separated by sepStr.
removeAt (str, num, sepStr)	Return str with the numth element separated by sepStr removed.
replaceAt (str, elemStr, num, sepStr)	Return str with the numth element separated by sepStr replaced with elemStr.
Conversion	
format (fmtStr, value)	Convert value to a string using the formatting string fmtStr.
toStr (value)	Return value converted to a string.

URL

Managing

escapeStr (str)	Return the escaped version of str.
getBase ()	Return the absolute URL of the current WMLScript compilation unit.
getReferer ()	Return the smallest URL of the resource that called the current compilation unit.
isValid (str)	Return true if str is a valid URL.
resolve (baseStr, embeddedStr)	Return a URL combining baseStr and embeddedStr.
unescapeStr (str)	Return the unescaped version of str.

Component

getFragment (str)	Return the URL fragment portion of str.
getHost (str)	Return the URL host portion of str.
getPort (str)	Return the URL server port num from str.
getParameters (str)	Return the URL parameters from str.
getPath (str)	Return the URL path from str.
getScheme (str)	Return the Internet protocol scheme from str.
getQuery (str)	Get the URL query portion from str.

Content

loadStr (urlStr, cTypeStr)	Return the content of type cTypeStr specified by urlStr.

WMLBrowser

Variables

getVar (str)	Return the value of the variable str from the current context.
setVar (str, value)	Set the value of the variable str to value.

Tasks

go (urlStr)	Queue deck urlStr for execution upon termination.
prev ()	Queue a prev task for execution upon termination.
refresh ()	Signal the WML user agent to do a <refresh>.
newContext ( )	Clear the current WML user agent context.

Query

getCurrentCard ( )	Return the smallest possible URL of the current WML user agent card.

Dialogs

prompt (str, defaultStr)	Display str and prompt for input using defaultStr as a default value.
confirm (str, okStr, cancelStr)	Display str, okStr, and cancelStr, and return true if okStr is selected.
alert (str)	Display str and wait for user confirmation.

נספח ה

התקליטור המצורף

בתקליטור המצורף לספר זה תוכל למצוא מספר דברים :

- קטלוג HTML – קטלוג ספרי המחשבים האינטראקטיבי של הוצאת הוד-עמי. הקטלוג מאפשר קריאת פרקים לדוגמה, תוכן עניינים, מגה-אינדקס ועוד.
- לשם קריאת הפרקים לדוגמה יש להתקין את תוכנת Adobe Acrobat Reader אשר מצורפת בתקליטור. הוראות התקנה בהמשך. הקטלוג מומלץ לצפייה באמצעות Internet Explorer 5.x ומעלה, המצורפת בתקליטור. הוראות התקנה בהמשך. התקנת שתי התוכנות קלה וניתנת לביצוע באמצעות קישור ישירות מהקטלוג.
- מספר תוכנות עזר שימושיות.
- **קטעי קוד המופיעים בספר.**



הערה !

אם מנהל התקן כונן התקליטורים המותקן הוא 16 סיביות - ייתכן שתראה רק שמונה תווים ראשונים של שם הקובץ (במקרה ובמקור הוא ארוך יותר).

הסיבה : כונני תקליטורים במהירות x4 עובדים עם מנהל התקן שעבד בסביבת DOS ו-Windows 3.11 ויכול לעבוד גם עם Windows 9x, למעט היכולת לזהות קבצים עם שמות ארוכים.

הפתרון : להתקין מנהל התקן 32 סיביות (אם קיים), או לקנות כונן תקליטורים חדש ולוודא שמצורף אליו מנהל התקן 32 סיביות.

אם אינך בטוח כיצד להתקין את התוכניות שבתקליטור פנה לאיש מקצוע.

קרא את קובץ ONCD שבתקליטור כדי לקבל עוד מידע לגבי התקליטור

התיקיה הרלוונטית לספר זה היא Books\59313

התקנת התקליטור המצורף

בתיקיה **X:\books\59313** (החלף את האות X באות הכונן המתאימה) תמצא את קטעי הקוד המופיעים בספר, מחולקים בתת-תיקיות לפי מספרי הפרקים.

בתיקיות אחרות בתיקיה Books נמצאים קבצים הרלוונטיים לספרים אחרים של ההוצאה.

העתקת קבצי המקור
לכונן הקשיח במחשב שלך

1. בחר בתפריט **התחל** > **תוכניות** > **סייר Windows**.
2. הצג את תוכן התיקיה Books אשר בתקליטור.
3. סמן את התיקיה 59313, וגרור אותה לתיקיה כלשהי בדיסק.

מכיון שמקור הקבצים הוא התקליטור, הם מסומנים **לקריאה בלבד**. רצוי לשנות מאפיין זה כדי שתוכל להשתמש בקבצים:

1. דרך הסייר היכנס לתיקיה בדיסק, שבה נמצאים הקבצים שהעתקת.
2. סמן קובץ מסוים או את כולם על ידי **Ctrl+A**.
3. הצב את סמן העכבר מעל האזור המסומן, ולחץ לחיצה ימנית בעכבר.
4. מתפריט הקיצור בחר **מאפיינים**.
5. בטל את הסימון בתיבה **קריאה בלבד**.
6. לחץ על **החל** ועל **אישור**.

Acrobat Reader - התקנה

יש להתקין תוכנה זו כדי לקרוא ולהדפיס את הפרקים לדוגמה, אליהם ניתן לגשת באמצעות קטלוג **HTML** (שהתקנתו תוסבר בהמשך). התוכנה גם מאפשרת חיפוש בעברית ובאנגלית במסמך המוצג. בנוסף, בעזרת תוכנה זו תוכל לקרוא את המסמכים שההוצאה מפרסמת באתר האינטרנט. התוכנה פועלת במערכות הפעלה **Windows 95 ומעלה**.

1. לחץ על לחצן **התחל** ובחר באפשרות **הפעלה**.
2. בתיבת הטקסט הקלד את הפקודה:
X:\Software\Adobe Acrobat\ARME4Heb.exe (החלף את האות X באות המייצגת את כונן התקליטורים שלך), ולחץ על **אישור**.

3. אשף ההתקנה מתקין את הרכיבים הנדרשים. עליך ללחוץ על **Next**, **Accept** ו-**Next** פעם נוספת, כדי לבצע את ההתקנה.

4. בסיום ההתקנה עשויה להופיע על המסך תיבת דו-שיח **התנגשות בין גירסאות** ומייד אחר כך להיעלם. במקומה תופיע על המסך תיבת הודעה של תוכנית ההתקנה. לחץ **אישור** ובתיבת הדו-שיח **התנגשות בין גירסאות** ששבה להופיע לחץ **כן**, כדי לשמור את גרסת הקובץ שלך.

קטלוג HTML

הוצאת הוד-עמי גאה לבשר על קטלוג HTML העושה שימוש בטכנולוגיות אינטרנט מתקדמות, כדי להביא לך את המידע על ספרי המחשבים המקצועיים שלנו בלחיצת לחצן העכבר.

התקנת הקטלוג אינה מעתיקה קבצים לכונן הדיסק הקשיח המקומי, אלא רק מציבה סמל קיצור דרך אל הקטלוג על שולחן העבודה שלך. להפעלת הקטלוג חייב התקליטור להימצא בכונן.

הקטלוג מומלץ לצפייה באמצעות **Microsoft Internet Explorer**.

בעזרת קטלוג HTML תוכל:

1. לעיין במידע אודות ספרי ההוצאה מתי שתרצה (לחיצה כפולה.... וזהו!).
2. לעבור במהירות ובקלות בין הקטלוג והיישום בו אתה עובד.
3. לעיין במידע אודות כל ספר וספר.
4. לצפות ואף להדפיס פרק לדוגמה.
5. לצפות ואף להדפיס מגה-אינדקס של הספר.
6. לגשת במהירות, בגישה אינטואיטיבית, תוך התמקדות מהירה בספר המבוקש.
7. לעיין בקטלוג בקצב אישי שלך.
8. לנווט את דרכך בקטלוג ולחזור ולהתרענן בכל נושא בכל רגע.

הקטלוג ניתן לצפייה באמצעות Internet Explorer

1. הכנס את התקליטור לכונן.
2. לחץ על **התחל** ובחר **הפעלה**.
3. בעזרת לחצן **עיון** סמן את הקובץ **Setup.exe** אשר בתיקיה הראשית של התקליטור המצורף.
4. לחץ **פתח**.
5. לחץ **אישור**.

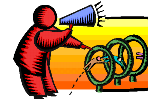
קטלוג ומחירון מעודכנים של ספרי ההוצאה נמצאים באתר האינטרנט www.hod-ami.co.il



1. ודא שתקליטור הוד-עמי נמצא בכונן התקליטורים.
2. הפעל את הסמל עם הכיתוב **קטלוג ספרי מחשבים בהוצאת הוד-עמי** שעל שולחן העבודה.

מה עוד בתקליטור?

הוצאת הוד-עמי מפיצה תוכנות אלו כבונוס ללקוחות ההוצאה, ואינה מתיימרת לגבות תשלום עבור התוכניות המצורפות, ו/או לתמוך בהם.



אזהרה!

השימוש בתוכן תקליטור זה הוא על אחריותו הבלעדית של המשתמש. המוצרים המותקנים בתקליטור זה מסופקים באחריות החברות המייצרות אותם, בהתאם לתנאי האחריות המצורפים לכל אחד מהמוצרים. הוצאת הוד-עמי אינה אחראית, בכל צורה שהיא, לאופן ולטיב התוכנות המותקנות.

בכל שאלה לגבי תוכנה הנמצאת בתקליטור, יש לפנות למפתחי התוכנה (כל תוכנה בנפרד), כפי שמצוין בקבצי העזרה של התוכנה המדוברת.

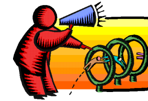
הקבצים הם גרסאות שיתופיות (ShareWare) וחופשיות (FreeWare). גירסה שיתופית (ShareWare) מאפשרת לך, המשתמש, לבדוק את יעילות התוכנה ואת תאימותה לעבודה אותה אתה מבצע. אם התוכנה נמצאה מתאימה לצרכיך, עליך לשלם למפתחיה תשלום סמלי (לפי הרשום בקבצי העזרה של כל תוכנה ותוכנה בנפרד) כדי לקבל רישיון מלא לשימוש בה. רכישת רישיון לשימוש בתוכנה תפתח בפניך מיגוון אפשרויות, שייתכן ולא עמדו לרשותך בהפעלת הגירסה השיתופית.

התקנת תוכנת גלישה לאינטרנט Microsoft Internet Explorer 5.5

תוכנית ההתקנה מזהה את גרסת מערכת ההפעלה ומתקינה את גרסת הדפדפן הדרושה.

ניתן להתקנה בכל מחשב בו מותקנת מערכת הפעלה Windows 95, Windows 98, Windows Me, Windows 2000 בממשק משתמש עברי או אנגלי.

1. הכנס את התקליטור לכונן.
2. לחץ על לחצן **התחל** ובחר באפשרות **הפעלה**.
3. לחץ על לחצן **עיון**.
4. בחר בכונן התקליטורים בתיקיה **Software\IE55** ובקובץ בשם **Setup.exe**.
5. לחץ על לחצן **פתח**.
6. לחץ על לחצן **אישור**.
7. פעל לפי ההוראות על המסך.



אזהרה!

לפני ביצוע שדרוג מ- Windows 95 ל- Windows 98 בעברית (זו בה התפריטים בעברית ולחצן התחל מימין שורת המשימות), יש להסיר כל התקנה קיימת של Internet Explorer 5. לאחר ביצוע השדרוג ניתן לבצע התקנה מחדש, של הגירסה המתאימה.

תיקיה ראשית SoftWare
(הרשימה חלקית ועשויה להשתנות)

בדרך כלל לחיצה כפולה על שם הקובץ המפורט ברשימה מפעילה את תוכנית ההתקנה.

שם תוכנה	תיאור	קובץ הפעלה	מבצע
Adobe Acrobat	תוכנה לצפייה בקבצי pdf, כולל תמיכה בעברית	ARME4Heb.exe	התקנה
ICQ	תוכנה לתקשורת אישית באינטרנט	ICQ2000b.exe	התקנה
MIRC	תוכנת הצ׳אט הפופולרית ביותר ברשת	mirc582t.exe	התקנה
NetEx	תוכנה המאפשרת גלישה בעברית	netex200.exe	התקנה
Power Toys	תוכניות שירות עבור Windows 95	W9xPowerToys.exe	פריסה
Power Toys	תוכניות שירות עבור Internet Explorer	IEPowerToys.exe	פריסה
WinAmp	תוכנה להשמעת קבצי MP3 (מוסיקה)	WinAmp25e.exe	התקנה
WinZip	תוכנית לפריסה/דחיסה של קבצים	WinZip80.exe	התקנה

אינדקס

A

B

C